

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:

Ryoko FUJIKAWA, et al.

Application No.: Unassigned

Group Art Unit: Unassigned

Filed: September 25, 2003

Examiner: Unassigned

For: PROGRAM AND PROCESS FOR GENERATING DATA USED IN SOFTWARE
FUNCTION TEST

**SUBMISSION OF CERTIFIED COPY OF PRIOR FOREIGN
APPLICATION IN ACCORDANCE
WITH THE REQUIREMENTS OF 37 C.F.R. § 1.55**

Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

Sir:

In accordance with the provisions of 37 C.F.R. § 1.55, the applicants submit herewith a certified copy of the following foreign application:

Japanese Patent Application No. 2002-278806

Filed: September 25, 2002

It is respectfully requested that the applicants be given the benefit of the foreign filing date as evidenced by the certified papers attached hereto, in accordance with the requirements of 35 U.S.C. § 119.

Respectfully submitted,

STAAS & HALSEY LLP

Date: September 25, 2003

By: 

William F. Herbert
Registration No. 31,024

1201 New York Ave, N.W., Suite 700
Washington, D.C. 20005
Telephone: (202) 434-1500
Facsimile: (202) 434-1501

日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日

Date of Application:

2002年 9月25日

出 願 番 号

Application Number:

特願2002-278806

[ST.10/C]:

[JP2002-278806]

出 願 人

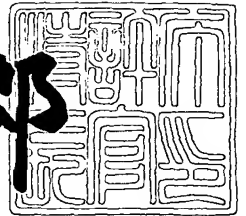
Applicant(s):

富士通株式会社

2003年 3月 4日

特 許 庁 長 官
Commissioner,
Japan Patent Office

太田 信一郎



出証番号 出証特2003-3013497

【書類名】 特許願

【整理番号】 0252045

【提出日】 平成14年 9月25日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 11/28

【発明の名称】 ソフトウェア機能テストデータ生成プログラムおよびソフトウェア機能テストデータ生成方法

【請求項の数】 5

【発明者】

 【住所又は居所】 神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

 【氏名】 藤川 亮子

【発明者】

 【住所又は居所】 神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内

 【氏名】 伊藤 吾朗

【発明者】

 【住所又は居所】 大分県大分市東春日町17番58号 株式会社富士通大分ソフトウェアラボラトリ内

 【氏名】 片山 徹也

【発明者】

 【住所又は居所】 静岡県静岡市南町18番1号 株式会社富士通インフォソフトテクノロジー内

 【氏名】 富永 泰伸

【発明者】

 【住所又は居所】 静岡県静岡市南町18番1号 株式会社富士通インフォソフトテクノロジー内

 【氏名】 川島 悟

【発明者】

【住所又は居所】 静岡県静岡市南町18番1号 株式会社富士通インフォ
ソフトテクノロジー内

【氏名】 齋藤 斎

【発明者】

【住所又は居所】 静岡県静岡市南町18番1号 株式会社富士通インフォ
ソフトテクノロジー内

【氏名】 長谷川 真則

【特許出願人】

【識別番号】 000005223

【氏名又は名称】 富士通株式会社

【代理人】

【識別番号】 100092152

【弁理士】

【氏名又は名称】 服部 毅巖

【電話番号】 0426-45-6644

【手数料の表示】

【予納台帳番号】 009874

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9705176

【ブルーフの要否】 要

【書類名】 明細書

【発明の名称】 ソフトウェア機能テストデータ生成プログラムおよびソフトウェア機能テストデータ生成方法

【特許請求の範囲】

【請求項 1】 開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援プログラムにおいて、

コンピュータに、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述から、前記更新後動作記述と共通性の高い前記更新前動作記述を選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する、

処理を実行させることを特徴とするテスト支援プログラム。

【請求項 2】 前記更新後動作記述と選択された前記更新前動作記述とで共通した情報の属性に応じて、抽出した前記更新前テストパターンから引き継ぐべき情報を決定することを特徴とする請求項 1 記載のテスト支援プログラム。

【請求項 3】 前記更新後テストパターンのうち、前記更新前テストパターンから引き継がない情報を自動生成することを特徴とする請求項 1 記載のテスト支援プログラム。

【請求項 4】 共通性の高い前記更新前動作記述がない場合、操作入力により指定された前記更新前動作記述を選択することを特徴とする請求項 1 記載のテスト支援プログラム。

【請求項 5】 開発対象プログラムで定義された機能に対して入力するパラ

メタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援方法において、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述から、前記更新後動作記述と共通性の高い前記更新前動作記述を選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する、

ことを特徴とするテスト支援方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は開発中のソフトウェアの機能を検証するための動作テストを支援するためのテスト支援プログラムおよびテスト支援方法に関し、特に機能に対して入力するパラメタと機能からの戻り値とを含むテストパターンを用いてテストを行う際のテストパターンの生成を支援するテスト支援プログラムおよびテスト支援方法に関する。

【0002】

【従来の技術】

大規模なソフトウェアの開発では、最初から不具合（バグ）の全くないプログラムを作成するのはほとんど不可能である。そのため、ソフトウェアを開発するには、作成したプログラムが設計通りに機能することをテストする必要がある。完成前のテストによって不具合を発見し、適宜プログラムを修正することにより、設計通りに正しく動作するソフトウェアを開発することが出来る。

【0003】

たとえば、オブジェクト指向で作成されたプログラムでは、機能毎にメソッドが作成される。メソッドは、入力データに応じて所定の処理を実行し、戻り値を出力する。そこで、オブジェクト指向のプログラムのテストでは、メソッドに対して入力するデータ（パラメタ）と、その入力に応じて出力されるべき戻り値との組からなるテストパターンを作成する。メソッドに処理を実行させたときに、テストパターンと同じ入出力が行われるか否かにより、正しく動作しているかどうかを検証できる。

【 0 0 0 4 】

ところが、ソフトウェア開発では、テストに必要な工数が非常に大きい。特に品質を重視する場合、プログラミング—プログラムテスト工程においてはプログラミング 2 : テスト 8 の割合を占めることもある。

【 0 0 0 5 】

そこで、ソフトウェア開発における大きな課題であるテストの効率化のために、テスト支援ツールが開発されている。テスト支援ツールとしては、ドライバ、スタブおよびデバッガ制御コマンド列を自動生成するものがある（例えば、特許文献 1 参照）。

【 0 0 0 6 】

ここで、ドライバは、被試験モジュールを呼び出すモジュールである。スタブは、被試験モジュールの下位モジュールの動作を擬似するモジュールである。デバッガ制御コマンドは、被試験モジュールの実行に必要な変数の入力データの設定個所の特定、および実行文中で実際に使用されるフィールドの特定を行い、入力データの設定、結果の確認を促すものである。

【 0 0 0 7 】

【特許文献 1】

特開平 6 - 2 5 0 8 8 4 号公報

【 0 0 0 8 】

【発明が解決しようとする課題】

しかし、テスト支援ツールで生成可能なテストパターンは、テストに必要な最低限のものである。そのため、信頼性の高いテストを行うには、ソフトウェアの

機能や処理対象として想定される情報に応じて、パラメタと戻り値との様々な組み合わせのテストパターンを追加で用意する必要がある。追加で用意するテストパターンは、ソフトウェアの機能に合わせて、作業者が適宜手入力で設定しなければならない。

【0009】

しかも、信頼性の高いプログラムテストを行うと、大量（たとえば、平均して10件/KStep（1000ステップ中の障害発生件数））の障害が検出される。そして、各障害の内容に応じて、障害修正や仕様変更が行われる。そのため、プログラムの変更内容に合わせたテストパターンの変更作業も頻繁に行われる。

【0010】

このように、従来は、開発中のソフトウェアのプログラムに変更が加えられる度に煩雑なテストパターン変更作業を行っているため、テストパターンの作成に多大な労力が費やされていた。また、テスト支援ツールを利用してもテストパターンの作成負荷が軽減されないということが、テスト支援ツールの適用阻害要因ともなっている。

【0011】

本発明はこのような点に鑑みてなされたものであり、テスト対象のプログラムの内容が更新された場合に、更新後のプログラム用のテストパターンを効率的に作成できるテスト支援プログラムおよびテスト支援方法を提供することを目的とする。

【0012】

【課題を解決するための手段】

本発明では上記課題を解決するために、図1に示すような処理をコンピュータに実行させるテスト支援プログラムが提供される。本発明に係るテスト支援プログラムは、開発対象プログラムで定義された機能に対して入力するパラメタと機能からの戻り値とを含むテストパターンを用いて、開発対象プログラムの動作テストを実施するためのものである。このテスト支援プログラムに従って、コンピュータが以下の処理を実行する。

【0013】

コンピュータは、開発対象プログラムの内容が更新されたとき、更新後の開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述 1 を取得する（ステップ S 1）。次に、コンピュータは、更新前の開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述 2 から、更新後動作記述と共通性の高い更新前動作記述 2 a を選択する（ステップ S 2）。さらに、コンピュータは、更新前の開発対象プログラムの動作テストのために作成された複数の更新前テストパターン 3 から、選択された更新前動作記述 2 a の動作テストのための更新前テストパターン 3 a を抽出する（ステップ S 3）。そして、コンピュータは、抽出した更新前テストパターン 3 a の少なくとも一部を引き継いで、更新後動作記述の動作テストのための更新後テストパターン 4 を生成する（ステップ S 4）。

【0014】

これにより、更新後動作記述 1 と共通性の高い更新前動作記述 2 a に対して設定された更新前テストパターン 3 a の内容を引き継いで、更新後動作記述 1 に対する更新後テストパターン 4 が生成される。

【0015】

また、本発明では、上記課題を解決するために、開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援方法において、前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述 1 を取得し（ステップ S 1）、更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述 2 から、前記更新後動作記述と共通性の高い前記更新前動作記述 2 a を選択し（ステップ S 2）、更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターン 3 から、選択された前記更新前動作記述の動作テストのための前記更新前テストパターン 3 a を抽出し（ステップ S 3）、抽出した前記更新前テストパターン 3 a の少なくとも一部を引き継いで、前記更新後動作記述 1 の動作テストのための更新後テ

ストパターン 4 を生成する（ステップ S 4）、ことを特徴とするテスト支援方法が提供される。

【0016】

これにより、更新後動作記述 1 と共通性の高い更新前動作記述 2 a に対して設定された更新前テストパターン 3 a の内容を引き継いで、更新後動作記述 1 に対する更新後テストパターン 4 が生成される。

【0017】

【発明の実施の形態】

以下、本発明の実施の形態を図面を参照して説明する。

まず、実施の形態に適用される発明の概要について説明し、その後、実施の形態の具体的な内容を説明する。

【0018】

図 1 は、実施の形態に適用される発明の概念図である。本発明では、開発対象プログラムで定義された機能に対して入力するパラメタと機能からの戻り値とを含むテストパターンを用いて開発対象プログラムの動作テストを実施する。そのために、まず、開発対象プログラムの内容が更新されたとき、更新後の開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述 1 を取得する（ステップ S 1）。更新後動作記述 1 は、たとえば、更新後の開発対象プログラムの内容を解析することで取得することができる。なお、更新後の開発対象プログラムに複数の機能が含まれていれば、機能毎の複数の更新後動作記述が生成されることとなる。

【0019】

次に、更新前の開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述 2 から、更新後動作記述と共通性の高い更新前動作記述 2 a を選択する（ステップ S 2）。なお、複数の更新後動作記述 2 は、たとえば、更新前の開発対象プログラムの内容を解析することで取得された機能毎の動作記述である。

【0020】

さらに、更新前の開発対象プログラムの動作テストのために作成された複数の

更新前テストパターン 3 から、ステップ S 2 で選択された更新前動作記述の動作テストのための更新前テストパターン 3 a を抽出する（ステップ S 3）。

【0 0 2 1】

そして、抽出した更新前テストパターン 3 a の少なくとも一部を引き継いで、更新後動作記述 1 の動作テストのための更新後テストパターン 4 を生成する（ステップ S 4）。

【0 0 2 2】

これにより、更新後動作記述 1 と共通性の高い更新前動作記述 2 a に対して設定された更新前テストパターン 3 a の内容を引き継いで、更新後動作記述 1 に対する更新後テストパターン 4 が生成される。その結果、開発対象プログラムが更新されたときにもテストパターンを一から作り直す必要が無くなり、テストパターンの作成が容易となる。テストパターンを容易に作成できることで、テスト支援プログラムを用いた動作テストを効率よく行うことができる。

【0 0 2 3】

以下、本発明の実施の形態を具体的に説明する。なお、以下の実施の形態では、オブジェクト指向言語のクラスにおける機能（メソッド）に着目して、オブジェクト指向で作成されたソフトウェアのテストを実行するテスト支援装置の例を示す。本発明をオブジェクト指向によるソフトウェア開発に適用することで、オブジェクト指向言語で書かれたソフトウェアプログラムの生産性を向上させることができる。

【0 0 2 4】

オブジェクト指向言語でソフトウェア開発を行う場合、テスト対象のクラスの機能が変更されたとき、設計済みのテストパターンを、変更後のメソッドに対応するテストパターンに引き継がせる。オブジェクト指向言語としては、主に j a v a（登録商標）や C ++ などがあるが、以下の実施の形態では、J a v a（登録商標）言語のテスト支援装置の例を示す。

【0 0 2 5】

オブジェクト指向言語の場合、機能（メソッド）はメソッドパラメタ値の入力パターンにより異なる動作をする。テスト支援ツールではこのメソッドを呼び出

す際のパラメタ値を様々な入力テストパターンとして設計する。その入力テストパターンを用いてメソッドを呼び出すことで、テストを実行できる。

【 0 0 2 6 】

ところで、オブジェクト指向言語によるソフトウェア開発では、クラスの仕様の変更、障害対応による変更などでメソッドのインタフェースの仕様が変更になることが開発中には多々ある。その場合、定義済みのテストパターン（メソッドの呼び出しパラメタの値）が無駄になりテストの効率が悪くなってしまう。そこで、そのような効率の悪化を防ぐために、クラスの仕様変更時にもメソッドにおけるテストパターンを自動で引き継ぐ機構を、以下に示すテスト支援装置によって実現する。

【 0 0 2 7 】

なお、以下の実施の形態では、機能（メソッド）のインタフェースの属性に着目し、一定のルールで新機能が旧機能に変更されたかを自動判断しテストパターンを引継ぐ機構と、判断できない場合には引継ぎ可能なテストパターンを利用者に提示する機構とを提供する。着目するインタフェースとしては「メソッド名（機能の名前）」、「メソッドパラメタ（機能の動作条件）」、「メソッド戻り値（機能の実行結果）」などが挙げられる。着目した属性には優先順位が付けられ、一致した属性の優先順位に基づいたルールにより、変更前メソッドと変更後メソッドの対応付けが自動で行われ、テストパターンの自動引継ぎが行われる。

【 0 0 2 8 】

まず、テスト支援装置のハードウェア構成について説明する。

図 2 は、本発明の実施の形態に用いるテスト支援装置のハードウェア構成例を示す図である。テスト支援装置 1 0 0 は、CPU (Central Processing Unit) 1 0 1 によって装置全体が制御されている。CPU 1 0 1 には、バス 1 0 7 を介して RAM (Random Access Memory) 1 0 2、ハードディスクドライブ (HDD: Hard Disk Drive) 1 0 3、グラフィック処理装置 1 0 4、入力インタフェース 1 0 5、および通信インタフェース 1 0 6 が接続されている。

【 0 0 2 9 】

RAM 1 0 2 には、CPU 1 0 1 に実行させる OS (Operating System) のプロ

グラムやアプリケーションプログラムの少なくとも一部が一時的に格納される。また、RAM 1 0 2 には、CPU 1 0 1 による処理に必要な各種データが格納される。HDD 1 0 3 には、OS やアプリケーションプログラムが格納される。

【0 0 3 0】

グラフィック処理装置 1 0 4 には、モニタ 1 1 が接続されている。グラフィック処理装置 1 0 4 は、CPU 1 0 1 からの命令に従って、画像をモニタ 1 1 の画面に表示させる。入力インタフェース 1 0 5 には、キーボード 1 2 とマウス 1 3 とが接続されている。入力インタフェース 1 0 5 は、キーボード 1 2 やマウス 1 3 から送られてくる信号を、バス 1 0 7 を介して CPU 1 0 1 に送信する。

【0 0 3 1】

通信インタフェース 1 0 6 は、ネットワーク 1 0 に接続されている。通信インタフェース 1 0 6 は、ネットワーク 1 0 を介して、他のテスト支援装置との間でデータの送受信を行う。

【0 0 3 2】

以上のようなハードウェア構成によって、本実施の形態のテスト支援装置の機能を実現することができる。

図 3 は、テスト支援装置の有する処理機能を示すブロック図である。図 3 に示すように、テスト支援装置 1 0 0 は、テストクラス記憶部 1 1 0、構文・意味解析部 1 2 0、第 1 の解析結果記憶部 1 3 0、テストパターン作成部 1 4 0、テストパターン記憶部 1 5 0、第 2 の解析結果記憶部 1 6 0、更新時自動引継ぎ部 1 7 0、テスト実行部 1 8 0 および手動引継ぎ部 1 9 0 を有している。またテスト支援装置 1 0 0 には、モニタ 1 1 やキーボード 1 2 等の入出力装置 1 4 が接続されている。

【0 0 3 3】

テストクラス記憶部 1 1 0 は、テストの対象となるクラス（テストクラス）を記憶する。クラスは、共通の属性（特性）を有するインスタンス（オブジェクト）の集合である。クラスは、個々のインスタンスを定義するひな形と考えることができる。クラスには、そのクラスに属する各インスタンスが共通にもつ属性項目の集合と手続きの集合がまとめて記述される。

【 0 0 3 4 】

構文・意味解析部 1 2 0 は、テストクラスの構文やその意味を解析し、解析結果情報を生成する。構文・意味解析部 1 2 0 は、パーサとも呼ばれる。構文・意味解析部 1 2 0 は、新規に作成されたテストクラスの解析結果情報を、第 1 の解析結果記憶部 1 3 0 に格納する。また、構文・意味解析部 1 2 0 は、内容が更新されたテストクラスの解析結果情報を、第 2 の解析結果記憶部 1 6 0 に格納する。

【 0 0 3 5 】

第 1 の解析結果記憶部 1 3 0 は、新規に作成されたテストクラスの解析結果を記憶する。

テストパターン作成部 1 4 0 は、第 1 の解析結果記憶部 1 3 0 に格納された解析結果情報を基にテストパターンを作成する。テストパターン作成部 1 4 0 は、作成したテストパターンをテストパターン記憶部 1 5 0 に格納する。

【 0 0 3 6 】

テストパターン記憶部 1 5 0 は、テストパターン作成部 1 4 0 で作成されたテストパターン、および更新時自動引継ぎ部 1 7 0 や更新時手動引継ぎ部 1 9 0 で引き継がれたテストパターンを記憶する。

【 0 0 3 7 】

第 2 の解析結果記憶部 1 6 0 は、既存のテストクラスから変更されたテストクラスの解析結果を記憶する。

更新時自動引継ぎ部 1 7 0 は、テストパターン記憶部 1 5 0 に格納されたテストパターンを基に、第 2 の解析結果記憶部 1 6 0 内の解析結果情報に応じたテストパターンを作成する。更新時自動引継ぎ部 1 7 0 は、作成したテストパターンを、テストパターン記憶部 1 5 0 に格納する。

【 0 0 3 8 】

テスト実施部 1 8 0 は、テストパターン記憶部 1 5 0 に格納されているテストパターンに基づいて、テストクラス記憶部 1 1 0 に格納されているテストクラスの動作テストを行う。

【 0 0 3 9 】

具体的には、テスト実施部180は、テストドライバ生成部181、テストスタブ生成部182、自動実行部183、および結果検証部184を有している。テストドライバ生成部181は、テストドライバを生成する。テストドライバは、被試験モジュールを呼び出すプログラムモジュールである。テストスタブ生成部182は、テストスタブを生成する。テストスタブは、試験対象のテストクラスの下位構造のクラスの動作を擬似的に実現するプログラムである。自動実行部183は、テストドライバ、テストスタブ、テストパターン等を用いて、テストクラスのテストを実行する。

【0040】

手動引継ぎ部190は、新規に作成されたテストクラスのテストパターンから、そのテストクラスの更新後のテストクラスのテストパターンへ、操作入力にตอบสนองしてテストパターンを引き継がせる。

【0041】

以上のような構成によって、テストパターンの自動引き継ぎが可能となる。

図4は、テストパターン自動引き継ぎの入出力関係を示す図である。更新前のテストクラス111は、構文・意味解析部120に入力される。すると、構文・意味解析部120により構文・意味の解析が行われ、更新前の解析結果情報131が生成される。生成された解析結果情報131は、テストパターン作成部140に入力される。テストパターン作成部140は、入力された解析結果情報131に基づいて、更新前のテストパターン151を作成する。テストパターン151は、テスト実施部180によって参照され、そのテストパターン151に基づいて更新前のテストクラス111の動作検証が行われる。

【0042】

更新前のテストクラス111に関する動作検証で、不具合が見付かった場合には、テストクラス111の内容が更新され、更新後のテストクラス112が生成される。更新後のテストクラス112は、構文・意味解析部120に入力される。すると、構文・意味解析部120により構文・意味の解析が行われ、更新後の解析結果情報161が生成される。

【0043】

その後、更新前の解析結果情報131と更新後の解析結果情報161とが更新時自動引継ぎ部170に入力される。更新時自動引継ぎ部170は、テストクラスの更新前後におけるそれぞれの解析結果情報131, 161を比較する。そして、更新時自動引継ぎ部170は、変更部分の引継ルールに従い引き継ぐことができるテストパターンを決定する。次に、更新時自動引継ぎ部170は、更新前のテストパターン151から、更新後のテストクラスに対する動作テストに引き継ぐことができるテストパターンを抽出する。そして、更新時自動引継ぎ部170は、抽出したテストパターンを、更新後のテストパターン152に設定する。

【0044】

更新時自動引継ぎ部170によるテストパターンの自動的な引き継ぎが行われた後、手動引継ぎ部190によって、更新前のテストパターン151と更新後のテストパターン152との内容が、入出力装置14の画面に比較表示される。このとき、自動引継ができなかった部分について、強調表示される。そして、作業者がキーボード12やマウス13を操作して、テストパターンの引き継ぎ指示を入力すると、その指示に従って手動引継ぎ部190が更新前のテストパターン151内のテストパターンを、更新後のテストパターン152に設定する。このようにして生成された更新後のテストパターン152に基づいて、テスト実施部180による動作検証が行われる。

【0045】

ところで、テストクラス111, 112は、様々なメソッド（機能）を有している。メソッドは、テストクラス111, 112を記述したソースプログラム内の一部の動作記述によって、その動作が定義されている。メソッドは、複数の情報が含まれており、各情報は属性を有している。

【0046】

図5は、メソッドに含まれる情報の属性を示す図である。これはjava（登録商標）言語定義でのメソッド20の構造を表している。図5に示すように、メソッド20は、修飾子21、メソッド戻り値22、メソッド名23、およびメソッドパラメタ24の属性の情報を有している。

【0047】

図 6 は、解析結果情報の保存形式例を示す図である。解析結果情報 1 3 1 は、複数のメソッド情報テーブル 1 3 1 a と複数のメソッドパラメータテーブル 1 3 1 b とで構成されている。メソッド情報テーブル 1 3 1 a には、メソッド ID、クラス名、メソッド名、メソッド戻り値、パラメータ数などの情報が設定される。メソッドパラメータテーブル 1 3 1 b には、パラメータ ID、メソッド ID、パラメータ名、パラメータ型などの情報が設定される。

【 0 0 4 8 】

メソッド情報テーブル 1 3 1 a とメソッドパラメータテーブル 1 3 1 b とは、メソッド ID によって関係づけられている。互いに関係するメソッド情報テーブル 1 3 1 a に登録されたメソッド情報とメソッドパラメータテーブル 1 3 1 b に登録されたメソッドパラメータとは、1 対 N (N は 0 以上の整数) の関係で存在している。すなわち、1 つのメソッド情報に対して、1 以上のメソッドパラメータが関係づけられる。

【 0 0 4 9 】

以下、Employee クラスをテストクラスとして、動作テストを行う場合の例について具体的に説明する。

図 7 は、更新前のテストクラス例を示す図である。この例では、テストクラス 1 1 1 において、J a v a (登録商標) 言語によるメソッドが記述されている。

【 0 0 5 0 】

第 1 のメソッド 1 1 1 a は、従業員名からの従業員検索機能を定義したメソッドである。第 2 のメソッド 1 1 1 b は、従業員氏名、年齢からの従業員検索機能を定義したメソッドである。第 3 のメソッド 1 1 1 c は、従業員氏名、年齢、部門コードからの従業員検索機能を定義したメソッドである。第 4 のメソッド 1 1 1 d は、従業員氏名、部門コードからの従業員検索機能を定義したメソッドである。第 5 のメソッド 1 1 1 e は、従業員コードからの従業員情報の更新機能を定義したメソッドである。第 6 のメソッド 1 1 1 f は、従業員コードからの従業員情報の更新機能を定義したメソッドである。

【 0 0 5 1 】

このようなテストクラス 1 1 1 が構文・意味解析部 1 2 0 で解析され、解析結

果情報が生成される。以下、EmployeeクラスのメソッドsearchEmployeebyNoの解析結果情報131の保存例を示す。図4で示したように、解析結果情報は、メソッド情報テーブルとメソッドパラメータテーブルとで構成される。

【0052】

図8は、更新前のメソッド情報テーブルの例を示す図である。メソッド情報テーブル131aの各欄の横方向に並べられた情報同士が互いに関連づけられ、関連づけられた情報が1つのメソッド情報を構成している。

【0053】

メソッドIDの欄に、第1のメソッド111a～第6のメソッド111fに対応する各メソッド情報に対して順番に1～6の番号が設定されている。クラス名の欄には、全てのメソッド情報に対してEmployeeと設定されている。メソッド名の欄には、各メソッド情報のメソッド名が設定されている。メソッドID「1」～「4」のメソッド名は「searchEmployee」である。メソッドID「5」のメソッド名は「updateEmployee」である。メソッドID「6」のメソッド名は「deleteEmployee」である。戻り値型の欄には、メソッドの戻り値の型が設定されている。メソッドID「1」～「4」の戻り値の型は「EmployeeData」である。メソッドID「5」の戻り値の型は「boolean」である。メソッドID「6」の戻り値の型は「int」である。

【0054】

パラメータ数の欄には、関連づけられたメソッドパラメータの数が設定されている。メソッドID「1」の第1のメソッド111aには、1つのメソッドパラメータが関連づけられている。メソッドID「2」の第2のメソッド111bには、2つのメソッドパラメータが関連づけられている。メソッドID「3」の第3のメソッド111cには、3つのメソッドパラメータが関連づけられている。メソッドID「4」の第4のメソッド111dには、2つのメソッドパラメータが関連づけられている。メソッドID「5」の第5のメソッド111eには、3つのメソッドパラメータが関連づけられている。メソッドID「6」の第6のメソッド111fには、1つのメソッドパラメータが関連づけられている。

【0055】

図9は、更新前のメソッドパラメータテーブルの例を示す図である。メソッドパラメータテーブル131bの各欄の横方向に並べられた情報同士が互いに関連づけられ、関連づけられた情報が1つのメソッドパラメータを構成している。

【0056】

パラメータIDの欄には、各メソッドパラメータに対して順番に1～12の番号が設定されている。

メソッドIDの欄には、各メソッドパラメータが関連づけられているメソッド情報のメソッドIDが設定されている。パラメータID「1」のメソッドパラメータは、メソッドID「1」のメソッド情報に関連づけられている。パラメータID「2」、「3」のメソッドパラメータは、メソッドID「2」のメソッド情報に関連づけられている。パラメータID「4」～「6」のメソッドパラメータは、メソッドID「3」のメソッド情報に関連づけられている。パラメータID「7」、「8」のメソッドパラメータは、メソッドID「4」のメソッド情報に関連づけられている。パラメータID「9」～「11」のメソッドパラメータは、メソッドID「5」のメソッド情報に関連づけられている。パラメータID「12」のメソッドパラメータは、メソッドID「6」のメソッド情報に関連づけられている。

【0057】

パラメータ名の欄には、各メソッドパラメータの名称が設定されている。パラメータID「1」、「2」、「4」、「7」、「9」のパラメータ名は「EmployeeName」である。パラメータID「3」、「5」、「10」のパラメータ名は「age」である。パラメータID「6」、「8」、「11」のパラメータ名は「code」である。パラメータID「12」のパラメータ名は「EmployeeNo」である。

【0058】

パラメータ型の欄には、各メソッドパラメータのデータ型が設定されている。メソッドID「1」、「2」、「4」、「7」、「9」のパラメータ型は「String」である。メソッドID「3」、「5」、「6」、「8」、「10」～「12」のパラメータ型は「int」である。

【0059】

このような解析結果情報に基づいて、テストパターン作成部140によりテス

トパターンが作成される。なお、テストパターン作成部140は、テストパターンの全てを自動で作成するのではなく、作業からの操作入力に基づいたテストパターンの生成も行う。例えば、テストパターン作成部140は、メソッド毎に、正常な戻り値が得られるテストパターンと正常な戻り値が得られないテストパターンとを生成する。そして、テストパターン作成部140は、入出力装置14による作業からの操作入力に応答して、様々なバリエーションのテストパターンを生成する。このように、作業が任意に入力したテストパターンを用いれば、信頼性の高い動作テストが可能である。

【0060】

図10～図14に、メソッド毎のテストパターンの例を示している。なお、以下の説明において、メソッド毎のテストパターンの集合をテストパターン群と呼ぶこととする。

【0061】

図10は、テストクラス変更前の第1のメソッドのテストパターンの例を示す図である。第1のメソッド111aのテストパターン群151aでは、各テストパターンに対して、識別番号（No.）、メソッド名、パラメタ、正常／異常、戻り値、およびテストの観点の欄が設けられている。各欄の横方向に並べられた情報同士が互いに関連づけられ、テストパターンを構成している。

【0062】

識別番号の欄には、テストパターンの識別番号が設定されている。第1のメソッド111aに対するテストパターンのグループにグループ番号M1が設定され、個々のテストパターンの識別番号は、グループ番号M1と枝番とで表される。図10の例では、3つのテストパターンに「M1-1」～「M1-3」の識別番号が設定されている。

【0063】

メソッド名の欄には、テストパターンがテストの対象としているメソッドの名称が設定される。図10の例は第1のメソッド111aに対するテストパターンであるため、全てのテストパターンに対して第1のメソッド111aの「search Employee」のパラメタが設定されている。このパラメタは、従業員の氏名を表す

パラメタである。

【0064】

パラメタの欄には、第1のメソッド111aに対して、パラメタ「EmployeeName」として入力する値が設定されている。識別番号「M1-1」のテストパターンでは、「F通 太郎」という値が入力される。識別番号「M1-2」のテストパターンでは、空白の値が入力される。識別番号「M1-3」のテストパターンでは、Null（空を示す情報）が入力される。

【0065】

正常／異常の欄には、入力するパラメタが正常な値なのか、異常な値なのかを示す情報が設定される。識別番号「M1-1」のテストパターンで入力されるパラメタは正常な値である。識別番号「M1-2」のテストパターンで入力されるパラメタは異常な値である。識別番号「M1-3」のテストパターンで入力されるパラメタは異常な値である。

【0066】

戻り値の欄には、第1のメソッド111aからの戻り値「EmployeeData」の種類を示す情報が設定される。識別番号「M1-1」のテストパターンを入力したときの戻り値は、検索結果である。識別番号「M1-2」のテストパターンを入力したときの戻り値は、Nullである。識別番号「M1-3」のテストパターンを入力したときの戻り値は、Nullである。

【0067】

テストの観点の欄は、テストパターンによって実行されるテストの目的が示されている。たとえば、識別番号「M1-1」のテストパターンは正常系のテストである。識別番号「M1-2」のテストパターンは、氏名が空白の場合の異常系のテストである。識別番号「M1-3」のテストパターンは、氏名がnullの場合の異常系のテストである。なお、テストの観点の欄は、テストパターンの意味を説明するために設けられた項目であり、テストパターンの自動引継やテスト実行のために必須の情報ではない。

【0068】

他のメソッドに対するテストパターンについては、第1のメソッド111aの

テストパターンと異なる部分についてのみ説明する。

図 1 1 は、テストクラス変更前の第 2 のメソッドのテストパターンの例を示す図である。第 2 のメソッド 1 1 1 b のテストパターン群 1 5 1 b では、グループ番号 M 2 が設定されている。図 1 2 の例では、「M 2 - 1」～「M 2 - 7」の 7 つのテストパターンが設定されている。メソッド 1 1 1 b のメソッド名は、「searchEmployee」である。

【 0 0 6 9 】

メソッド 1 1 1 b のテストパターンでは、「EmployeeName」と「Age」とのパラメタが設定される。「Age」は従業員の年齢を示すパラメタである。

この例では、正常系のテストパターン（識別番号「M 2 - 1」）や従業員の氏名に関する異常系のテストパターン（識別番号「M 2 - 2」、識別番号「M 2 - 3」）に加え、従業員の年齢に関する異常系のテストパターンが設定されている。たとえば、年齢が 0 の場合の異常系（識別番号「M 2 - 4」）、年齢が従業員雇用範囲外の場合（小さすぎと大きすぎ）の異常系（識別番号「M 2 - 5」、識別番号「M 2 - 6」）、年齢が null の場合の異常系（識別番号「M 2 - 7」）である。

【 0 0 7 0 】

図 1 2 は、テストクラス変更前の第 3 のメソッドのテストパターンの例を示す図である。第 3 のメソッド 1 1 1 c のテストパターン群 1 5 1 c では、グループ番号 M 3 が設定されている。図 1 2 の例では、「M 3 - 1」～「M 3 - 1 0」の 1 0 個のテストパターンが設定されている。メソッド 1 1 1 c のメソッド名は、「searchEmployee」である。

【 0 0 7 1 】

メソッド 1 1 1 c のテストパターンでは、「EmployeeName」、「Age」、および「code」のパラメタが設定される。「code」は各従業員を一意に識別するための従業員コードを示すパラメタである。

【 0 0 7 2 】

この例では、正常系のテストパターン（識別番号「M 3 - 1」）や従業員の氏名や年齢に関する異常系のテストパターン（識別番号「M 3 - 2」～識別番号「

M3-7」)に加え、従業者の従業員コードに関する異常系のテストパターンが設定されている。たとえば、従業員コードが0の場合の異常系(識別番号「M3-8」)、従業員コードがNullの場合の異常系(識別番号「M3-9」)である。なお、この例では、全てのパラメタがnullの場合の異常系(識別番号「M3-10」)に関するテストパターンも設定されている。

【0073】

図13は、テストクラス変更前の第4のメソッドのテストパターンの例を示す図である。第4のメソッド111dのテストパターン群151dでは、グループ番号M4が設定されている。図13の例では、「M4-1」～「M4-6」の6個のテストパターンが設定されている。メソッド111dのメソッド名は、「searchEmployee」である。メソッド111dのテストパターンでは、「EmployeeName」と「code」とのパラメタが設定される。

【0074】

この例では、正常系のテストパターン(識別番号「M4-1」)や従業者の氏名や従業員コードに関する異常系のテストパターン(識別番号「M4-2」～識別番号「M4-6」)が設定されている。

【0075】

図14は、テストクラス変更前の第5のメソッドのテストパターンの例を示す図である。第5のメソッド111eのテストパターン群151eでは、グループ番号M5が設定されている。図14の例では、「M5-1」～「M5-10」の10個のテストパターンが設定されている。メソッド111eのメソッド名は、「UpdateEmployee」である。メソッド111eのテストパターンでは、「EmployeeName」、「Age」、および「code」のパラメタが設定される。

【0076】

この例では、図12に示した第3のメソッド111cのテストパターンと同様に、正常系のテストパターン(識別番号「M5-1」)や従業者の氏名、年齢、従業員コードに関する異常系のテストパターン(識別番号「M5-2」～識別番号「M5-10」)が設定されている。

【0077】

図15は、テストクラス変更前の第6のメソッドのテストパターンの例を示す図である。第6のメソッド111fのテストパターン群151fでは、グループ番号M6が設定されている。図15の例では、「M6-1」～「M6-4」の4個のテストパターンが設定されている。メソッド111fのメソッド名は、「deleteEmployee」である。メソッド111fのテストパターンでは、「code」のパラメタが設定される。

【0078】

この例では、正常系のテストパターン（識別番号「M6-1」）や従業員コードに関する異常系のテストパターン（識別番号「M6-2」～識別番号「M6-4」）が設定されている。従業員コードの異常系には、従業員コードが0の場合、従業員コードがnullの場合、従業員コードが該当なしの場合がある。なお、正常系の戻り値は「True」であり、異常系の戻り値は「False」である。

【0079】

このようなテストパターンを用いて、テストクラスが設計通りに動作するかどうかをテストすることができる。

なお、テストパターン作成部140で自動作成できないテストパターンとして、たとえば、第2のメソッド111bの識別番号「M2-5」や識別番号「M2-6」のテストパターン、第6のメソッド111fの識別番号「M6-4」のテストパターンなどがある。第2のメソッド111bの識別番号「M2-5」や識別番号「M2-6」のテストパターンは、従業員雇用範囲である年齢層が分かっていると設定できない。また、第6のメソッド111fの識別番号「M6-4」のテストパターンは、該当者の以存在しない従業員コードが分かっていると設定できない。そのため、これらのテストパターンについては、テストクラスを運用するときの環境に合わせたパラメタを作業者が入力する必要がある。

【0080】

このようなテストの設定情報（テストパターンの集合）を機能（メソッド）が変更になった場合も引き継いで利用できれば、テストクラスの運用環境に合わせて作業者が設定したテストクラスの再設定が不要となる。そこで、本実施の形態では、機能の変更パターンとその場合の引継ぎルールを決めることにより上記設

定情報を新しい機能（メソッド）でも利用可能とする。

【0081】

ここで、引き継ぎのルールとして、機能（メソッド）の属性に着目する。上記の例では、3つの属性がある。機能の実現に不可欠な属性であるこの3属性に優先順位をつける。そして、更新前のメソッドと更新後のメソッドとの間で、属性の一致／不一致によって、テストパターンの引き継ぎの可否を決定する。更新前のメソッドといずれかの属性が一致する更新後のメソッドが複数有れば、高優先度の属性が一致する更新後のメソッドにテストパターンを引き継がせる。

【0082】

本実施の形態における各属性の優先順位は、以下の通りである。

第1優先 メソッド名（機能を表す表現）

第2優先 メソッドパラメタ（機能が動作するときの条件）

第3優先 メソッド戻り値（機能の結果）

ここで、各属性を引き継ぎ可否の判定基準に採用した理由について説明する。

【0083】

・第1の優先をメソッド名とした理由

オブジェクト指向プログラミングでは、異なるメソッドに同じメソッド名が与えられることない。また、一般的に、メソッドの内容が部分的に変更されてもメソッド名が変更されることはない。そのため、更新前のテストクラス111と更新後のテストクラス112とにおいて、メソッド名が同一のメソッドであれば、そのメソッドのテストパターンにおけるメソッドパラメタと戻り値とを引き継げる。したがって、本実施の形態では、メソッド名を第1優先の属性としている。

【0084】

・第2の優先をメソッドパラメタとした理由

テストクラスに対するテストでは、メソッドに対して正常なパラメタが入力されたときに正しい戻り値が返され、メソッドに対して異常なパラメタが入力されたときにエラー等の異常を表す戻り値が返されることが検証される。テストパターンとしては、正常なパラメタの組を含むものと、少なくとも一部に異常なパラメタを含むパラメタの組を含むものとが用意される。メソッドを更新する場合、

入力されるパラメタが同じままで、動作内容に変更が加えられる場合も多い。そのため、更新前のテストクラス111と更新後のテストクラス112とにおいて、入力されるメソッドパラメタの組が同じであれば、そのメソッドのテストパターンのメソッドパラメタを引き継げる。したがって、本実施の形態では、メソッドパラメタ名を第2優先の属性としている。

【0085】

・第3の優先をメソッド戻り値とした理由

更新前のテストクラス111と更新後のテストクラス112とにおいて、メソッドパラメタが異なっても、メソッド戻り値が同じであれば、メソッド戻り値を引き継げる。そこで、本実施の形態では、メソッド戻り値を第3優先の属性としている。

【0086】

以下、テストクラス更新時のテストパターンの引き継ぎ状況について説明する。

図16は、更新後のテストパターンの例を示す図である。図7に示した更新前のテストパターンからの変更部分は以下の通りである。

【0087】

第1のメソッド112aは、修飾子に変更されている。更新前の第1のメソッド111aにおいて「public」であった修飾子が、更新後の第1のメソッド112aでは「private」となっている。

【0088】

第2のメソッド112bは、メソッド戻り値に変更されている。更新前の第2のメソッド111bにおいて「EmployeeData」であったメソッド戻り値が、更新後の第2のメソッド112bでは「EmployeeData[]」となっている。

【0089】

第3のメソッド112cは、メソッドパラメタに変更されている。更新前の第3のメソッド111cにおいて「String EmployeeName,int Age,int code」であったメソッドパラメタが、更新後の第3のメソッド112cでは「String EmployeeName,long Age,int section,Boolean retire」となっている。

【0090】

第4のメソッド112dは、メソッド戻り値とメソッドパラメタとが変更されている。更新前の第4のメソッド111dにおいて「EmployeeData」であったメソッド戻り値が、更新後の第4のメソッド112dでは「EmployeeData[]」となっている。更新前の第3のメソッド111dにおいて「String EmployeeName, int code」であったメソッドパラメタが、更新後の第4のメソッド112dでは「String EmployeeName, int section」となっている。

【0091】

第5のメソッド112eは、メソッド名が変更されている。更新前の第5のメソッド111eにおいて「updateEmployee」であったメソッド名が、更新後の第5のメソッド112eでは「updateEmployeebyNo」となっている。

【0092】

第6のメソッド112fは、メソッド名とメソッドパラメタとが変更されている。更新前の第6のメソッド111fにおいて「deleteEmployee」であったメソッド名が、更新後の第6のメソッド112fでは「deleteEmployeebyNo」となっている。更新前の第6のメソッド111fにおいて「int EmployeeNo」であったメソッドパラメタが、更新後の第6のメソッド112fでは「int EmployeeNo, String EmployeeName」となっている。

【0093】

このようなメソッド112a～112fを含む更新後のテストクラス112が生成されると、構文・意味解析部120において解析が行われ、更新後の解析結果情報161が生成される。

【0094】

図17は、更新後のメソッド情報テーブルの例を示す図である。メソッド情報テーブル161aの各欄の横方向に並べられた情報同士が互いに関連づけられ、関連づけられた情報が1つのメソッド情報を構成している。

【0095】

メソッドIDの欄に、各メソッド情報に対して順番に1～6の番号が設定されている。クラス名の欄には、全てのメソッド情報に対してEmployeeと設定されて

いる。

【0096】

メソッド名の欄には、各メソッド情報のメソッド名が設定されている。メソッドID「1」～「4」のメソッド名は「searchEmployee」である。メソッドID「5」のメソッド名は「updateEmployeeNo」である。メソッドID「6」のメソッド名は「deleteEmployeeNo」である。

【0097】

戻り値型の欄には、メソッドの戻り値の型が設定されている。メソッドID「1」、「3」の戻り値の型は「EmployeeData」である。メソッドID「2」、「4」の戻り値の型は「EmployeeData[]」である。メソッドID「5」の戻り値の型は「Boolean」である。メソッドID「6」の戻り値の型は「int」である。

【0098】

パラメタ数の欄には、関連づけられたメソッドパラメタの数が設定されている。メソッドID「1」のメソッドには、1つのメソッドパラメタが関連づけられている。メソッドID「2」のメソッドには、2つのメソッドパラメタが関連づけられている。メソッドID「3」のメソッドには、4つのメソッドパラメタが関連づけられている。メソッドID「4」のメソッドには、2つのメソッドパラメタが関連づけられている。メソッドID「5」のメソッドには、3つのメソッドパラメタが関連づけられている。メソッドID「6」のメソッドには、1つのメソッドパラメタが関連づけられている。

【0099】

図18は、更新後のメソッドパラメタテーブルの例を示す図である。メソッドパラメタテーブル161bの各欄の横方向に並べられた情報同士が互いに関連づけられ、関連づけられた情報が1つのメソッドパラメタを構成している。

【0100】

パラメタIDの欄には、各メソッドパラメタに対して順番に1～14の番号が設定されている。

メソッドIDの欄には、各メソッドパラメタが関連づけられているメソッド情報のメソッドIDが設定されている。パラメタID「1」のメソッドパラメタは

、メソッドID「1」のメソッド情報に関連づけられている。パラメタID「2」、「3」のメソッドパラメタは、メソッドID「2」のメソッド情報に関連づけられている。パラメタID「4」～「7」のメソッドパラメタは、メソッドID「3」のメソッド情報に関連づけられている。パラメタID「8」、「9」のメソッドパラメタは、メソッドID「4」のメソッド情報に関連づけられている。パラメタID「10」～「12」のメソッドパラメタは、メソッドID「5」のメソッド情報に関連づけられている。パラメタID「13」、「14」のメソッドパラメタは、メソッドID「6」のメソッド情報に関連づけられている。

【0101】

パラメタ名の欄には、各メソッドパラメタの名称が設定されている。パラメタID「1」、「2」、「4」、「8」、「10」、「14」のパラメタ名は「EmployeeName」である。パラメタID「3」、「5」、「11」のパラメタ名は「Age」である。パラメタID「6」、「9」のパラメタ名は「Section」である。パラメタID「7」のパラメタ名は「Retire」である。パラメタID「12」のパラメタ名は「Code」である。パラメタID「13」のパラメタ名は「Employee No」である。

【0102】

パラメタ型の欄には、各メソッドパラメタのデータ型が設定されている。メソッドID「1」、「2」、「4」、「8」、「10」、「14」のパラメタ型は「String」である。メソッドID「3」、「6」、「9」、「11」～「13」のパラメタ型は「int」である。メソッドID「5」のパラメタ型は「long」である。メソッドID「7」のパラメタ型は「Boolean」である。

【0103】

図17、図18に示したような更新後の解析結果情報が生成されると、更新時自動引継ぎ部170により、テストパターン自動引継処理が行われる。図19～図21に、テストパターン自動引継処理のフローチャートを示す。

【0104】

図19は、テストパターン自動引継処理の基本的なメインのフローチャートである。以下、図19に示す処理をステップ番号に沿って説明する。なお、以下の

処理は、全て更新時自動引継ぎ部 170 が行う処理である。

【0105】

〔ステップ S11〕更新後の解析結果情報 161 から未処理のメソッド情報のレコードを 1 つ読み込む。具体的には、メソッド情報テーブル 161 a からメソッド情報を読み込み、読み込んだメソッド情報に関連づけられたメソッドパラメタのレコードをメソッドパラメタテーブル 161 b から読み込む。なお、メソッド情報とメソッドパラメタとは、メソッド ID によって関連づけられている。すなわち、同じメソッド ID を有するメソッド情報とメソッドパラメタとが関連づけられている。

【0106】

〔ステップ S12〕処理対象のメソッドが無くなった（メソッド終了）か否かを判断する。具体的には、ステップ S11 における読み込み処理で、読み込み対象となる未処理のメソッドが発見されなかったときに、メソッド終了と判断される。メソッド終了の場合には、テストパターン自動引継処理が終了する。メソッド終了でない（ステップ S11 でメソッド情報のレコードが読み込まれた）場合には、処理がステップ S13 に進められる。

【0107】

〔ステップ S13〕ステップ S11 で読み込んだメソッド情報と 3 属性の情報（メソッド名、パラメタ名とパラメタ型、および戻り値型）が一致する更新前のメソッド情報が、更新前の解析結果情報 131 内にあるか否かを判定する（第 1 の判定）。該当するメソッド情報があれば、処理がステップ S14 に進められる。該当するメソッド情報がなければ、処理がステップ S16 に進められる。

【0108】

〔ステップ S14〕ステップ S13 で該当すると判断されたメソッド情報に対応するテストパターンにおけるメソッドパラメタの情報を、ステップ S11 で読み込んだメソッド情報のテストパターンに引き継ぐ。

【0109】

〔ステップ S15〕ステップ S13 で該当すると判断されたメソッド情報に対応するテストパターンにおける戻り値の情報を、ステップ S11 で読み込んだメ

ソッド情報のテストパターンに引き継ぐ。その後、処置がステップ S 1 1 に進められる。

【0110】

【ステップ S 1 6】 ステップ S 1 1 で読み込んだメソッド情報と、戻り値以外の 2 属性の情報（メソッド名、およびパラメタ名とパラメタ型）が一致する更新前のメソッド情報が、更新前の解析結果情報 1 3 1 内にあるか否かを判定する（第 2 の判定）。該当するメソッド情報があれば、処理がステップ S 1 7 に進められる。該当するメソッド情報がなければ、処理がステップ S 1 9 に進められる。

【0111】

【ステップ S 1 7】 ステップ S 1 6 で該当すると判断されたメソッド情報に対応するテストパターンにおけるメソッドパラメタの情報を、ステップ S 1 1 で読み込んだメソッド情報のテストパターンに引き継ぐ。

【0112】

【ステップ S 1 8】 メソッド戻り値の情報を自動生成し、ステップ S 1 1 で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処理がステップ S 1 1 に進められる。

【0113】

【ステップ S 1 9】 ステップ S 1 1 で読み込んだメソッド情報と、パラメタ（パラメタ名とパラメタ型）以外の 2 属性の情報（メソッド名と戻り値型）が一致する更新前のメソッド情報が、更新前の解析結果情報 1 3 1 内にあるか否かを判定する（第 3 の判定）。該当するメソッド情報があれば、処理がステップ S 2 0 に進められる。該当するメソッド情報がなければ、処理がステップ S 2 2 に進められる。

【0114】

【ステップ S 2 0】 不一致パラメタ引継処理が行われる。この処理の詳細は後述する。

【ステップ S 2 1】 ステップ S 1 9 で該当すると判断されたメソッド情報に対応するテストパターンにおける戻り値の情報を、ステップ S 1 1 で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処置がステップ S 1 1 に進め

られる。

【0115】

【ステップS22】 ステップS11で読み込んだメソッド情報と、メソッド名のみが一致し、他の2属性の情報（パラメタ名とパラメタ型、および戻り値型）が不一致の更新前のメソッド情報が、更新前の解析結果情報131内にあるか否かを判定する（第4の判定）。該当するメソッド情報があれば、処理がステップS23に進められる。該当するメソッド情報がなければ、処理がステップS25に進められる。

【0116】

【ステップS23】 不一致パラメタ引継処理が行われる。この処理の詳細は後述する。

【ステップS24】 メソッド戻り値の情報を自動生成し、ステップS11で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処理がステップS11に進められる。

【0117】

【ステップS25】 ステップS11で読み込んだメソッド情報と、パラメタ（パラメタ名とパラメタ型）が一致し、メソッド名が不一致の更新前のメソッド情報が、更新前の解析結果情報131内にあるか否かを判定する（第5の判定）。該当するメソッド情報があれば、処理がステップS26に進められる。該当するメソッド情報がなければ、処理がステップS32に進められる。

【0118】

【ステップS26】 ステップS25の判断で該当したメソッド情報が1レコードだけか否かを判断する。1レコードだけ該当した場合には、処理がステップS27に進められる。複数のレコードが該当した場合には、処理がステップS28に進められる。

【0119】

【ステップS27】 ステップS25で該当すると判断されたメソッド情報に対応するテストパターンにおけるメソッドパラメタの情報を、ステップS11で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処理がステップS

29に進められる。

【0120】

【ステップS28】ステップS25で該当すると判断された複数のメソッド情報のうち、メソッド名が一致する文字数が最も多いメソッド情報を引継対象として選択する。そして、選択したメソッド情報に対応するテストパターンにおけるメソッドパラメタの情報を、ステップS11で読み込んだメソッド情報のテストパターンに引き継ぐ。

【0121】

【ステップS29】テストパラメタの引継元となったメソッド情報と、ステップS11で読み込んだメソッド情報との戻り値型が一致するか否かを判断する。戻り値型が一致する場合には、処理がステップS30に進められる。戻り値型が一致しない場合には、処理がステップS31に進められる。

【0122】

【ステップS30】テストパラメタの引継元となったメソッド情報に対応するテストパターンにおける戻り値の情報を、ステップS11で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処置がステップS11に進められる。

【0123】

【ステップS31】メソッド戻り値の情報を自動生成し、ステップS11で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処置がステップS11に進められる。

【0124】

【ステップS32】ステップS11で読み込んだメソッド情報へテストパターンを引き継がせる対象となる更新前のメソッド情報が無いと判定し、処理をステップS11に進める。

【0125】

図20は、不一致パラメタ引継処理を示すフローチャートである。以下、図20に示す処理をステップ番号に沿って説明する。なお、以下の処理は、全て更新時自動引継ぎ部170が行う処理である。

【 0 1 2 6 】

〔ステップ S 4 1〕 ステップ S 1 1 で読み込んだメソッド情報の未処理のパラメタを 1 つ読み込む。

〔ステップ S 4 2〕 処理対象のパラメタが無くなった（パラメタ終了）か否かを判断する。具体的には、ステップ S 4 1 における読み込み処理で、読み込み対象となる未処理のパラメタが発見されなかったときに、パラメタ終了と判断される。パラメタ終了の場合には、不一致パラメタ引継処理を終了する。パラメタ終了でない（ステップ S 4 1 でパラメタが読み込まれた）場合には、処理がステップ S 4 3 に進められる。

【 0 1 2 7 】

〔ステップ S 4 3〕 ステップ S 4 1 で読み込んだパラメタと、パラメタ名およびパラメタ型が一致するパラメタが、引継対象となっている更新前のメソッド情報に含まれるか否かを判断する（第 6 の判定）。該当するパラメタがある場合には、処理がステップ S 4 4 に進められる。該当するパラメタがない場合には、処理がステップ S 4 5 に進められる。

【 0 1 2 8 】

〔ステップ S 4 4〕 引継対象のメソッド情報に対するテストパターン内のステップ S 4 3 で該当すると判定されたパラメタの情報を、ステップ S 1 1 で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処理がステップ S 4 1 に進められる。

【 0 1 2 9 】

〔ステップ S 4 5〕 ステップ S 4 1 で読み込んだパラメタと、パラメタ名が一致するパラメタが、引継対象となっている更新前のメソッド情報に含まれるか否かを判断する（第 7 の判定）。該当するパラメタがある場合には、処理がステップ S 4 6 に進められる。該当するパラメタがない場合には、処理がステップ S 4 7 に進められる。

【 0 1 3 0 】

〔ステップ S 4 6〕 パラメタ名が一致し、パラメタ型が不一致の場合の引継処理を実行する。この処理の詳細は後述する。

【ステップS47】ステップS41で読み込んだパラメタと、パラメタ型が一致するパラメタが、引継対象となっている更新前のメソッド情報に含まれるか否かを判断する（第8の判定）。該当するパラメタがある場合には、処理がステップS48に進められる。該当するパラメタがない場合には、処理がステップS49に進められる。

【0131】

【ステップS48】引継対象のメソッド情報に対するテストパターン内のステップS47で該当すると判定されたパラメタの情報を、ステップS11で読み込んだメソッド情報のテストパターンに引き継ぐ。その後、処理がステップS41に進められる。

【0132】

【ステップS49】ステップS11で読み込んだメソッド情報のテストパターンのパラメタに、デフォルト値を設定する。その後、処理がステップS41に進められる。

【0133】

図21は、パラメタ名が一致しパラメタ型が不一致の場合の処理を示すフローチャートである。以下、図21に示す処理をステップ番号に沿って説明する。なお、以下の処理は、全て更新時自動引継ぎ部170が行う処理である。

【0134】

【ステップS61】パラメタ型を比較する。更新前のパラメタ型が数値であり更新後のパラメタ型が数値の場合、処理がステップS62に進められる。更新前のパラメタ型が数値であり更新後のパラメタ型が文字型の場合、処理がステップS63に進められる。更新前のパラメタ型がある種の文字型であり更新後のパラメタ型が別種の文字型の場合、処理がステップS64に進められる。その他の場合、処理がステップS65に進められる。

【0135】

【ステップS62】数値から数値への変更の場合、更新前のテストパターンのパラメタを、そのまま更新後のテストパターンに引き継がせる。その後、処理が図20のステップS41に進められる。

【0136】

【ステップS63】数値から文字型への変更の場合、更新前のテストパターンのパラメタのパラメタ型を文字型に変更して、更新後のテストパターンに引き継がせる。その後、処理が図20のステップS41に進められる。

【0137】

【ステップS64】ある種の文字型から別種の文字型への変更の場合、更新前のテストパターンのパラメタのパラメタ型を別種の文字型に変更して、更新後のテストパターンに引き継がせる。その後、処理が図20のステップS41に進められる。

【0138】

【ステップS65】その他の場合には、更新後のテストパターンのパラメタとしてデフォルト値を新規に生成する。その後、処理が図20のステップS41に進められる。

【0139】

このようにして、更新前のテストクラス111をテストするために作成されたテストパターン151の一部または全部が、更新後のテストクラス112をテストするためのテストパターン152に引き継がれる。

【0140】

なお、図19に示したフローチャートには、5つの判定処理（ステップS13，S16，S19，S22，S25）が含まれている。各判定処理では、更新前と更新後とのそれぞれのメソッドの属性の相異点を判定している。

【0141】

図22は、判定処理とメソッドの属性との関係を示す図である。図22には、判定処理毎に、判定結果がYESとなるためのメソッドの属性の一致／不一致の関係を示している。属性の一致を丸印（○）で示し、不一致をばつ印（×）で示している。

【0142】

第1の判定（ステップS13）では、メソッド名、パラメタ、戻り値が一致した場合に、判定結果がYESとなる。第2の判定（ステップS16）では、メソ

ッド名、パラメタが一致し、戻り値が不一致の場合に、判定結果がYESとなる。第3の判定（ステップS19）では、メソッド名、戻り値が一致し、パラメタが不一致の場合に、判定結果がYESとなる。第4の判定（ステップS22）では、メソッド名のみが一致し、パラメタと戻り値とが不一致の場合に、判定結果がYESとなる。第5の判定（ステップS25）では、パラメタが一致し、メソッド名が不一致の場合に、判定結果がYESとなる（戻り値の一致不一致は影響しない）。

【0143】

ここで、図7に示した更新前のテストクラス111が更新され、図16に示した更新後のテストクラス112になったときのテストパターン引継例を詳しく説明する。

【0144】

図23は、メソッドの変更に応じたテストパターンの自動反映結果の一例を示す図である。

更新後の第1のメソッド112aは、1番目の判定（ステップS13）において、更新前の第1のメソッド111aを引継対象のメソッドとしたときに、判定結果がYESとなる。その結果、図10に示したテストパターン群151a内の全てのテストパターン（識別番号「M1-1」～「M1-3」）の内容が引き継がれる。

【0145】

更新後の第2のメソッド112bは、2番目の判定（ステップS16）において、更新前の第2のメソッド111bを引継対象のメソッドとしたときに、判定結果がYESとなる。その結果、図11に示したテストパターン群151b内の全てのテストパターン（識別番号「M2-1」～「M2-7」）の内容が引き継がれる。

【0146】

更新後の第3のメソッド112cは、3番目の判定（ステップS19）において、更新前の第3のメソッド111cを引継対象のメソッドとしたときに、判定結果がYESとなる。その結果、図12に示したテストパターン群151c内の

全てのテストパターン（識別番号「M3-1」～「M3-10」）の内容が引き継がれる。

【0147】

更新後の第4のメソッド112dは、4番目の判定（ステップS22）において、更新前の第4のメソッド111dを引継対象のメソッドとしたときに、判定結果がYESとなる。その結果、図13に示したテストパターン群151d内の全てのテストパターン（識別番号「M4-1」～「M4-6」）の内容が引き継がれる。

【0148】

更新後の第5のメソッド112eは、5番目の判定（ステップS25）において、更新前の第5のメソッド111eを引継対象のメソッドとしたときに、判定結果がYESとなる。その結果、図14に示したテストパターン群151e内の全てのテストパターン（識別番号「M5-1」～「M5-10」）の内容が引き継がれる。

【0149】

更新後の第6のメソッド112fは、全ての判定において、引継対象のメソッドが検出されず、判定結果がNOとなる。その結果、更新後の第6のメソッド112fに関しては、テストパターンの自動引き継ぎは行われない。

【0150】

なお、テストパターンの内容が引き継がれるときであっても、必ず全ての内容が引き継がれるのではない。すなわち、パラメタと戻り値との何れかのみが引き継がれる場合もある。以下、図23に示したメソッドの判定結果毎の具体的なテストパターン引継例を説明する。

【0151】

・第1のメソッドのテストパターン引継例

更新後のテストクラス112の第1のメソッド112aは、引継対象として更新前のテストクラス111の第1のメソッド111aが選択される。更新前の第1のメソッド111aに対するテストパターン群151aは図10に示す通りである。

【0152】

ここで、変更前の第1のメソッド111aと変更後の第1のメソッド112aとを比較する。

図24は、第1のメソッドの修正内容を示す図である。図24に示すように、修飾子が「public」から「private」に変更されている。

【0153】

図25は、第1のメソッドの属性毎の比較結果を示す図である。図25に示すように、その他（修飾子）のみが変更されており、メソッド名、メソッドパラメタ、メソッド戻り値型については、変更されていない。

【0154】

この場合、更新前の第1のメソッド111aに対して設定されていたテストパターン群151a（図10に示す）がそのまま、更新後の第1のメソッド112aのテストパターン152として引き継がれる。

【0155】

・第2のメソッドのテストパターン引継例

更新後のテストクラス112の第2のメソッド112bは、引継対象として更新前のテストクラス111の第2のメソッド111bが選択される。更新前の第2のメソッド111bに対するテストパターン群151bは図11に示す通りである。

【0156】

ここで、変更前の第2のメソッド111bと変更後の第2のメソッド112bとを比較する。

図26は、第2のメソッドの修正内容を示す図である。図26に示すように、メソッド戻り値が「EmployeeData」から「EmployeeData[]」に変更されている。

【0157】

図27は、第2のメソッドの属性毎の比較結果を示す図である。図27に示すように、メソッド戻り値のみが変更されており、メソッド名、メソッドパラメタ、その他の属性については変更されていない。

【0158】

この場合、テストパターン群 151b 中の各テストパターンのうち、メソッドパラメタの情報が引き継がれ、メソッド戻り値にはデフォルト値が設定される。

図 28 は、更新後の第 2 のメソッドのテストパターンの例を示す図である。更新後の第 2 のメソッドのテストパターン群 152b 内の各テストパターンと図 11 に示した更新前のテストパターンとの違いは、戻り値がデフォルトの値になったことである。他の情報は、図 11 に示した更新前のテストパターンと同じである。図 28 の例では、メソッド戻り値のデフォルトの値は、「Null」である。

【0159】

・第 3 のメソッドのテストパターン引継例

更新後のテストクラス 112 の第 3 のメソッド 112c は、引継対象として更新前のテストクラス 111 の第 3 のメソッド 111c が選択される。更新前の第 3 のメソッド 111c に対するテストパターン群 151c は図 12 に示す通りである。

【0160】

ここで、更新前の第 3 のメソッド 111c と更新後の第 3 のメソッド 112c とを比較する。

図 29 は、第 3 のメソッドの修正内容を示す図である。図 29 に示すように、メソッドパラメタが「String EmployeeName,int Age,int code」から「String EmployeeName,long Age,int section,boolean retire」に変更されている。

【0161】

図 30 は、第 3 のメソッドの属性毎の比較結果を示す図である。図 30 に示すように、メソッドパラメタのみが変更されており、メソッド名、メソッドパラメタ、その他の属性については変更されていない。

【0162】

以下に、パラメタが不一致の場合の判定例を示す。

(第 6 の判定) パラメタ名とパラメタ型とが一致するものを引き継ぐ。図 30 に示した例によれば、「String EmployeeName」は、更新前後において変更がなく、パラメタ名とパラメタ型とが一致する。したがって、更新前のテストパターンにおける「String EmployeeName」に対応する情報は、そのままを更新後の

テストパターンに引き継がれる。

【0163】

(第7の判定) 第6の判定の処理で残存するパラメタに対し、パラメタの名前が一致し、データが引き継ぎ可能なものを引き継ぐ。ここで、データ引き継ぎ可能なものとは、パラメタ型が数値→数値の場合、数値→文字列の場合、ある種の文字列→別種の文字列の場合である。数値→数値の場合には、型変換可能な範囲で引き継ぐ(数字系の基本データ型 `int long double floate decimal BicDecimal`)。数字→文字列の場合には、数字を文字列に変換して引き継ぐ。文字列→文字列の場合には、そのまま引き継ぐ。

【0164】

図30に示した例では、更新後の「`long age`」のパラメタに対して、パラメタ名が一致し、パラメタ型が不一致の更新前の「`int age`」が検出される。この場合、同じ数字系の場合なので、数字として引き継ぐことができる。そこで、更新前のテストパターンにおいてパラメタ「`int age`」として設定されていた値が、更新後のテストパターンではパラメタ「`long age`」の値として引き継がれる。この場合、同じ数字系の場合なので、数字として引き継ぐ。

【0165】

(第8の判定) 第7の判定処理で残存するパラメタに対し、パラメタの型が一致するものを引き継ぐ。図30に示した例では、更新後の「`int section`」のパラメタに対して、パラメタの型が一致する更新前の「`int code`」が検出される。そこで、更新前のテストパターンにおいてパラメタ「`int code`」として設定されていた値が、更新後のテストパターンではパラメタ「`int section`」の値として引き継がれる。

【0166】

第8の判定処理で引き継ぎ可能な既存パラメタが見つからない場合は、新規パラメタのテストデータとして、デフォルト値の新規データを生成する。ここで、デフォルト値は、テストデータの入力初期値である。基本型数字の場合、自由な数値が与えられる。図30に示した例では、「`boolean retire`」は引き継ぎ対象がないためデフォルト値「`true`」が生成され、この値を用いてテストパターンが作

成される。

【0167】

図31は、更新後の第3のメソッドのテストパターンの例を示す図である。図31に示す更新後の第3のメソッドのテストパターン群152cでは、テストパターンのメソッド名は省略している。また、パラメタの比較判定結果を分かり易くするために、更新前のパラメタ、更新後のパラメタ、および比較結果を表示している。

【0168】

更新前のテストパターンにおけるパラメタ「String EmployeeName」の値は、そのまま更新後のテストパターンにおけるパラメタ「String EmployeeName」に引き継がれている。更新前のテストパターンにおけるパラメタ「int Age」の値は、そのまま更新後のテストパターンにおけるパラメタ「long Age」に引き継がれている。更新前のテストパターンにおけるパラメタ「int Code」の値は、そのまま更新後のテストパターンにおけるパラメタ「int Section」に引き継がれている。更新後のテストパターンにおけるパラメタ「Boolean Retire」は、引き継ぎ対象がないため、全てのテストパターンに対してデフォルト値「true」が設定されている。

【0169】

・第4のメソッドのテストパターン引継例

更新後のテストクラス112の第4のメソッド112dは、引継対象として更新前のテストクラス111の第4のメソッド111dが選択される。更新前の第4のメソッド111dに対するテストパターン群151dは図13に示す通りである。

【0170】

ここで、変更前の第4のメソッド111dと変更後の第4のメソッド112dとを比較する。

図32は、第4のメソッドの修正内容を示す図である。図32に示すように、メソッド戻り値が「EmployeeData」から「EmployeeData[]」に変更されていると共に、メソッドパラメタが「String EmployeeName,int code」から「String Emp

loyeeName,int section」に変更されている。

【0171】

図33は、第4のメソッドの属性毎の比較結果を示す図である。図33に示すように、メソッド戻り値とメソッドパラメタが変更されており、メソッド名、その他の属性については変更されていない。

【0172】

パラメタのみが不一致の場合と同様に、パラメタ名とパラメタ型との判定が行われ、パラメタ毎の引き継ぎ対象が決定される。

(第6の判定) パラメタ名とパラメタ型とが一致するパラメタ同士で引き継ぎが行われる。図33に示した例では、「String EmployeeName」は更新前後において変更がなく、パラメタ名とパラメタ型とが一致する。したがって、更新前のテストパターンにおける「String EmployeeName」に対応する値は、そのままを更新後のテストパターンに引き継がれる。

【0173】

(第7の判定) 第6の判定処理で残存するパラメタのうち、パラメタの名前が一致し、データが引き継ぎ可能なものを引き継ぐ。図33の例では、該当するパラメタはない。

【0174】

(第8の判定) 第8の判定処理で残存するパラメタは、パラメタの型が一致するものを引き継ぐ。図33の例では、更新前のテストパターンにおける「int code」に対応する値が、更新後のテストパターンにおける「int section」に引き継がれる。

【0175】

図34は、更新後の第4のメソッドのテストパターンの例を示す図である。図34に示す更新後の第4のメソッドのテストパターン群152dでは、テストパターンのメソッド名は省略している。また、パラメタの比較判定結果を分かり易くするために、更新前のパラメタ、更新後のパラメタ、および比較結果を表示している。

【0176】

更新前のテストパターンにおけるパラメタ「String EmployeeName」の値は、そのまま更新後のテストパターンにおけるパラメタ「String EmployeeName」に引き継がれている。更新前のテストパターンにおけるパラメタ「int Code」の値は、そのまま更新後のテストパターンにおけるパラメタ「int Section」に引き継がれている。更新後の各テストパターンのメソッド戻り値は、デフォルトの値（予め設定された初期値）である。

【0177】

・第5のメソッドのテストパターン引継例

更新後のテストクラス112の第5のメソッド112eは、引継対象として更新前のテストクラス111の第5のメソッド111eが選択される。更新前の第5のメソッド111eに対するテストパターン群151eは図14に示す通りである。

【0178】

ここで、変更前の第5のメソッド111eと変更後の第5のメソッド112eとを比較する。

図35は、第5のメソッドの修正内容を示す図である。図35に示すように、メソッド名が「updateEmployee」から「updateEmployeebyNo」に変更されている。

【0179】

図36は、第5のメソッドの属性毎の比較結果を示す図である。図36に示すように、メソッド名のみが変更されており、メソッドパラメタ、メソッド戻り値、その他の属性については変更されていない。

【0180】

この場合、パラメタが一致する更新前のメソッドが検索される。すると、更新前の第3のメソッド111cと第5のメソッド111eとが検出される。このように、パラメタの一致がNレコード（Nは2以上の整数）ある場合、名前の文字列一致が多いメソッドが引き継ぎ対象となる。この例では、第5のメソッド111eが引き継ぎ対象となる。したがって、更新前の第5のメソッド111eに対して設定されていたテストパターン（図14に示す）のメソッドパラメタの値が

、更新後の第5のメソッド1 1 2 eのテストパターンに引き継がれる。また、更新前の第5のメソッド1 1 1 eと更新後の第5のメソッド1 1 2 eとは、メソッド戻り値が一致している。そのため、更新前の第5のメソッド1 1 1 eのテストパターンのメソッド戻り値の値が、更新後の第5のメソッド1 1 2 eのテストパターンに引き継がれる。

【0 1 8 1】

図3 7は、更新後の第5のメソッドのテストパターンの例を示す図である。図3 7に示す更新後の第5のメソッドのテストパターン群1 5 2 eでは、パラメタの比較判定結果を分かり易くするために、更新前のパラメタ、更新後のパラメタ、および比較結果を表示している。図3 7に示すように、更新前のテストパターンにおけるパラメタ、戻り値が引き継がれ、メソッド名のみが更新後の第5のメソッド1 1 2 eのメソッド名に変更されている。

【0 1 8 2】

以上のようにして、更新前のテストクラス1 1 1をテストするために用意されたテストパターンを用いて、更新後のテストクラス1 1 2をテストするためのテストパターンを自動生成することができる。このとき、更新後のテストクラス1 1 2に含まれるメソッドのうち、テストパターンを自動生成できるのは、引き継ぎ対象となる更新前のメソッドが存在するものである。引き継ぎ対象のメソッドが検出されなかった更新後のメソッドについては、手動引き継ぎ部1 9 0を用いて設定することができる。すなわち、作業からの操作入力に応答して、手動引き継ぎ部1 9 0が更新前のメソッドに対して設定されていたテストパターンを、更新後のメソッドに引き継がせる。

【0 1 8 3】

本実施の形態では、更新後のメソッドに対して、引き継ぎ対象となる更新前のメソッドを対応付け、テストパターンを引き継がせるための操作入力画面が、GUI (Graphical User Interface)により提供される。

【0 1 8 4】

図3 8は、引き継ぎ対象メソッド指定画面の例を示す図である。図3 8に示すように、引き継ぎ対象メソッド指定画面2 0 0には、更新後のメソッド情報表示部2 1 0

、更新前のメソッド情報表示部 2 2 0、引継候補のないメソッド一覧表示部 2 3 0、上へボタン 2 0 1、下へボタン 2 0 2、削除ボタン 2 0 3、追加ボタン 2 0 4、OK ボタン 2 0 5、キャンセルボタン 2 0 6、およびヘルプボタン 2 0 7 が設けられている。

【 0 1 8 5 】

更新後のメソッド情報表示部 2 1 0 は、更新後のメソッド情報のリストを表示するための表示領域である。更新後のメソッド情報表示部 2 1 0 内には、更新後のクラス名やメソッド名が表示されている。

【 0 1 8 6 】

更新前のメソッド情報表示部 2 2 0 は、更新前のメソッド情報のうち、更新対象として選択されたメソッドのリストを表示するための表示領域である。更新前のメソッド情報表示部 2 2 0 内には、更新前のクラス名、メソッド名、ファイル名（テストパターンが登録されているファイル）などが表示されている。

【 0 1 8 7 】

更新後のメソッド情報表示部 2 1 0 と更新前のメソッド情報表示部 2 2 0 とは、横方向（同一行）に並べられたメソッド情報同士が互いに関連づけられている。互いに関連するメソッド情報同士で、テストパターンの引継が行われる。なお、更新後のメソッド情報の引継対象の更新前のメソッドが選択されていない場合、その更新後のメソッドと同じ行の更新前のメソッド情報表示部 2 2 0 内の欄が、空欄になっている。また、更新前のメソッド情報表示部 2 2 0 に表示されている更新前のメソッド情報の登録位置を変更することで、更新前のメソッド情報と更新後のメソッド情報との関連づけを変更することができる。

【 0 1 8 8 】

引継候補のないメソッド一覧表示部 2 3 0 は、更新前のメソッド情報のうち、更新対象として選択されていないメソッド情報（引継候補のないメソッド情報）のリストを表示するための表示領域である。引継候補のないメソッド情報の 1 つを選択して、更新前のメソッド情報表示部 2 2 0 内の空欄の位置に移動することで、引継候補のないメソッド情報を、引継対象のない更新後のメソッド情報に関連づけることができる。更新前のメソッド情報の引継候補のないメソッド一覧表

示部 2 3 0 から更新前のメソッド情報表示部 2 2 0 への移動は、ドラッグ&ドロップと呼ばれるマウス操作で指示することができる。

【0 1 8 9】

上へボタン 2 0 1 は、更新前のメソッド情報表示部 2 2 0 内で選択されている更新前のメソッド情報の登録位置を、上へ移動させるためのボタンである。上へボタン 2 0 1 が押されると、更新前のメソッド情報表示部 2 2 0 内で選択されている更新前のメソッド情報の位置が 1 つ上の段へ移動する。このとき、1 つ上の段に登録されていた更新前のメソッド情報は、1 つ下の段に移動する。

【0 1 9 0】

下へボタン 2 0 2 は、更新前のメソッド情報表示部 2 2 0 内で選択されている更新前のメソッド情報の登録位置を、下へ移動させるためのボタンである。下へボタン 2 0 2 が押されると、更新前のメソッド情報表示部 2 2 0 内で選択されている更新前のメソッド情報の位置が 1 つ下の段へ移動する。このとき、1 つ下の段に登録されていた更新前のメソッド情報は、1 つ上の段に移動する。

【0 1 9 1】

削除ボタン 2 0 3 は、更新前のメソッド情報表示部 2 2 0 内で選択されている更新前のメソッド情報の登録を抹消するためのボタンである。削除ボタン 2 0 3 が押されると、更新前のメソッド情報表示部 2 2 0 内で選択されている更新前のメソッド情報が、更新前のメソッド情報表示部 2 2 0 から削除される。更新前のメソッド情報表示部 2 2 0 から削除された更新前のメソッド情報は、引継候補のないメソッド一覧表示部 2 3 0 内に設定される。

【0 1 9 2】

追加ボタン 2 0 4 は、引継候補のないメソッド一覧表示部 2 3 0 内で選択されている更新前のメソッド情報を、更新前のメソッド情報表示部 2 2 0 内で選択されている欄に追加するためのボタンである。追加ボタン 2 0 4 が押されると、引継候補のないメソッド一覧表示部 2 3 0 内で選択されている更新前のメソッド情報が、更新前のメソッド情報表示部 2 2 0 内で選択されている欄に追加される。更新前のメソッド情報表示部 2 2 0 に追加された更新前のメソッド情報は、引継候補のないメソッド一覧表示部 2 3 0 から削除される。

【0193】

OKボタン205は、引継対象メソッド指定画面200内の情報を確定するためのボタンである。OKボタン205が押されると、手動引継ぎ部190により、引継対象メソッド指定画面200に設定されている内容がテストパターン記憶部150に登録される。

【0194】

キャンセルボタン206は、引継対象メソッド指定画面200内での情報の変更内容を取り消すためのボタンである。キャンセルボタン206が押されると、引継対象メソッド指定画面200内で行われた操作が取り消され、引継対象メソッド指定画面200が閉じる。

【0195】

ヘルプボタン207は、操作方法のヘルプ画面を表示させるためのボタンである。ヘルプボタン207が押されると、操作方法を案内するヘルプ画面が表示される。

【0196】

このように、自動引継ぎ結果の確認と引き継がれなかった情報の手動引継ぎ支持を目的に、作業者が引き継ぎ対象を選択するためのGUIが提供される。これにより、自動引継の対象となる更新前のメソッドが存在しない場合でも、簡単な操作で引継対象の更新前のメソッドを設定することができる。引継対象の更新前のメソッドが指定されれば、そのメソッドのテストパターンが、更新後のメソッドのテストパターンとして引き継がれる。

【0197】

また、本実施の形態では、テストパターンの自動引継結果の詳細を画面上で容易に確認することができる。

図39は、テストパターン引継結果確認画面の例を示す図である。テストパターン引継結果確認画面300には、テストパターン内容表示部310とパラメタ表示部320とが設けられている。

【0198】

テストパターン内容表示部310は、メソッド毎のテストパターンの内容を表

示するための表示領域である。テストパターン内容表示部 310 には、テストパターン毎に、テストパターン番号、テスト内容、正常／異常等の各種情報が表示されている。

【0199】

パラメタ表示部 320 は、各テストパターンのパラメタを表示するための表示領域である。パラメタ表示部 320 には、各パラメタのパラメタ名、属性、値等が表示されている。パラメタ表示部 320 において自動生成された値については、強調表示される。たとえば、自動生成された値は、他の値とは異なる色で表示される。図 39 の例では、パラメタ「param1」の値「10」が強調表示されている。

【0200】

このようにテストパターンの引継処理において自動生成された値が強調表示することで、作業者に対して、自動生成された値の確認を促すことができる。作業者は、強調表示された項目の内容を確認し、値の適否を判断する。より適当な値がある場合には、パラメタ表示部 320 に表示されている値を変更することで、パラメタの値を変更することができる。

【0201】

以上説明したように本実施の形態によれば、テスト対象のクラスの機能（メソッド）が変更された場合に、設計済みのテストパターンを変更後のメソッドに対応したテストパターンに引き継ぐことができる。

【0202】

しかも、機能（メソッド）のインタフェースの属性（メソッド名、メソッドパラメタ、メソッド戻り値など）に着目し、この属性の一致の優先順位のルールにより更新前メソッドと更新後のメソッドの対応づけを自動で行いテストパターンを自動で引き継いでいる。これにより、より適当なテストパターンを引き次ぐことができる。

【0203】

さらに、自動引継ぎの候補がなかった場合に、手動で利用者が引き継ぎ対象を指示できる GUI を備えていることにより、引継対象の割り当ての操作が容易と

なる。

【0204】

また、自動引継ぎした部分を利用者にわかりやすく通知するGUIを備えていることにより、自動引継の適否を確認するのが容易となる。

また、不一致部分の自動引継ぎ対象外のメソッドパラメタに関してはデフォルト値（初期値）を自動設定するようにしたため、自動引継をしたテストパターンに関しては、そのままの状態を更新後のプログラムの動作テストに利用することができる。

【0205】

この際、デフォルト値が設定されたパラメタを強調表示するようにしたため、デフォルト値が設定されたパラメタを容易に認識することができ、適当な他の値に変更するのが容易となる。

【0206】

なお、上記の処理機能は、コンピュータによって実現することができる。その場合、テスト支援装置が有すべき機能の処理内容を記述したテスト支援プログラムが提供される。そのプログラムをコンピュータで実行することにより、上記処理機能がコンピュータ上で実現される。処理内容を記述したプログラムは、コンピュータで読み取り可能な記録媒体に記録しておくことができる。コンピュータで読み取り可能な記録媒体としては、磁気記録装置、光ディスク、光磁気記録媒体、半導体メモリなどがある。磁気記録装置には、ハードディスク装置（HDD）、フレキシブルディスク（FD）、磁気テープなどがある。光ディスクには、DVD(Digital Versatile Disc)、DVD-RAM(Random Access Memory)、CD-ROM(Compact Disc Read Only Memory)、CD-R(Recordable)/RW(Rewritable)などがある。光磁気記録媒体には、MO(Magneto-Optical disc)などがある。

【0207】

プログラムを流通させる場合には、たとえば、そのプログラムが記録されたDVD、CD-ROMなどの可搬型記録媒体が販売される。また、プログラムをサーバコンピュータの記憶装置に格納しておき、ネットワークを介して、サーバコ

ンピュータから他のコンピュータにそのプログラムを転送することもできる。

【0208】

プログラムを実行するコンピュータは、たとえば、可搬型記録媒体に記録されたプログラムもしくはサーバコンピュータから転送されたプログラムを、自己の記憶装置に格納する。そして、コンピュータは、自己の記憶装置からプログラムを読み取り、プログラムに従った処理を実行する。なお、コンピュータは、可搬型記録媒体から直接プログラムを読み取り、そのプログラムに従った処理を実行することもできる。また、コンピュータは、サーバコンピュータからプログラムが転送される毎に、逐次、受け取ったプログラムに従った処理を実行することもできる。

【0209】

(付記1) 開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援プログラムにおいて、

コンピュータに、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述から、前記更新後動作記述と共通性の高い前記更新前動作記述を選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する、

処理を実行させることを特徴とするテスト支援プログラム。

【0210】

(付記2) 予め指定された複数の属性の情報の少なくとも1つが前記更新後動作記述と同じ値の前記更新前動作記述を、共通性の高い前記更新前動作記述と

して選択することを特徴とする付記 1 記載のテスト支援プログラム。

【 0 2 1 1 】

（付記 3） 前記複数の属性に優先順が設定されており、高い優先順の属性の情報が同一である前記更新前動作記述を優先的に選択することを特徴とする付記 2 記載のテスト支援プログラム。

【 0 2 1 2 】

（付記 4） 前記複数の属性として、機能を特定する機能名、機能に入力するパラメタの種別、機能が処理を実行後の出力する戻り値の種別を含むことを特徴とする付記 2 記載のテスト支援プログラム。

【 0 2 1 3 】

（付記 5） 前記更新後動作記述と選択された前記更新前動作記述とで共通した情報の属性に応じて、抽出した前記更新前テストパターンから引き継ぐべき情報を決定することを特徴とする付記 1 記載のテスト支援プログラム。

【 0 2 1 4 】

（付記 6） 前記複数の属性として、機能を特定する機能名、機能に入力するパラメタの種別、および機能が処理を実行後の出力する戻り値の種別が指定されており、機能名が一致した場合、他の同一属性に対応するデータを、前記更新前テストパターンから引き継ぐことを特徴とする付記 5 記載のテスト支援プログラム。

【 0 2 1 5 】

（付記 7） 前記更新後テストパターンのうち、前記更新前テストパターンから引き継がない情報を自動生成することを特徴とする付記 1 記載のテスト支援プログラム。

【 0 2 1 6 】

（付記 8） 前記自動生成した情報を強調して、前記更新後テストパターンの内容を表示することを特徴とする付記 7 記載のテスト支援プログラム。

（付記 9） 共通性の高い前記更新前動作記述がない場合、操作入力により指定された前記更新前動作記述を選択することを特徴とする付記 1 記載のテスト支援プログラム。

【 0 2 1 7 】

（付記 1 0） 選択されなかった前記更新前動作記述の一覧を表示し、操作入力により前記一覧内で指定された前記更新前動作記述を選択することを特徴とする付記 9 記載のテスト支援プログラム。

【 0 2 1 8 】

（付記 1 1） オブジェクト指向で作成された開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援プログラムにおいて、

コンピュータに、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後メソッド情報を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前メソッド情報から、前記更新後メソッド情報と共通性の高い前記更新前メソッド情報を選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前メソッド情報の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後メソッド情報の動作テストのための更新後テストパターンを生成する、

処理を実行させることを特徴とするテスト支援プログラム。

【 0 2 1 9 】

（付記 1 2） 開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援方法において、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述から、前記更新後動作記述と共通性の高い前記更新前動作記述を

選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する、

ことを特徴とするテスト支援方法。

【 0 2 2 0 】

(付記 1 3) 開発対象プログラムで定義された機能に対して入力するパラメタと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するテスト支援装置において、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得する更新後動作記述取得手段と、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述を記憶する更新前動作記述記憶手段と、

前記更新前動作記述記憶手段から、更新後動作記述取得手段で取得された前記更新後動作記述と共通性の高い前記更新前動作記述を選択する更新前動作記述選択手段と、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンを記憶する更新前テストパターン記憶手段と、

更新前テストパターン記憶手段から、前記更新前動作記述選択手段で選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出する更新前テストパターン抽出手段と、

前記更新前テストパターン抽出手段で抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する更新後テストパターン生成手段と、

を有することを特徴とするテスト支援装置。

【 0 2 2 1 】

(付記 1 4) 開発対象プログラムで定義された機能に対して入力するパラメータと前記機能からの戻り値とを含むテストパターンを用いて前記開発対象プログラムの動作テストを実施するためのテスト支援プログラムを記録したコンピュータ読み取り可能な記録媒体において、

前記コンピュータに、

前記開発対象プログラムの内容が更新されたとき、更新後の前記開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述を取得し、

更新前の前記開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述から、前記更新後動作記述と共通性の高い前記更新前動作記述を選択し、

更新前の前記開発対象プログラムの動作テストのために作成された複数の更新前テストパターンから、選択された前記更新前動作記述の動作テストのための前記更新前テストパターンを抽出し、

抽出した前記更新前テストパターンの少なくとも一部を引き継いで、前記更新後動作記述の動作テストのための更新後テストパターンを生成する、

処理を実行させることを特徴とするテスト支援プログラムを記録したコンピュータ読み取り可能な記録媒体。

【 0 2 2 2 】

【発明の効果】

以上説明したように本発明では、更新後動作記述と共通性の高い更新前動作記述に対して設定された更新前テストパターンの内容を引き継いで、更新後動作記述に対する更新後テストパターンを生成するようにしたため、開発対象プログラムが更新されたときにもテストパターンを一から作り直す必要が無くなり、テストパターンの作成が容易となる。

【図面の簡単な説明】

【図 1】

実施の形態に適用される発明の概念図である。

【図 2】

本発明の実施の形態に用いるコンピュータのハードウェア構成例を示す図であ

る。

【図 3】

テスト支援装置の有する処理機能を示すブロック図である。

【図 4】

テストパターン自動引き継ぎの入出力関係を示す図である。

【図 5】

メソッドに含まれる情報の属性を示す図である。

【図 6】

解析結果情報の保存形式例を示す図である。

【図 7】

更新前のテストクラス例を示す図である。

【図 8】

更新前のメソッド情報テーブルの例を示す図である。

【図 9】

更新前のメソッドパラメータテーブルの例を示す図である。

【図 1 0】

テストクラス変更前の第 1 のメソッドのテストパターンの例を示す図である。

【図 1 1】

テストクラス変更前の第 2 のメソッドのテストパターンの例を示す図である。

【図 1 2】

テストクラス変更前の第 3 のメソッドのテストパターンの例を示す図である。

【図 1 3】

テストクラス変更前の第 4 のメソッドのテストパターンの例を示す図である。

【図 1 4】

テストクラス変更前の第 5 のメソッドのテストパターンの例を示す図である。

【図 1 5】

テストクラス変更前の第 6 のメソッドのテストパターンの例を示す図である。

【図 1 6】

更新後のテストパターンの例を示す図である。

【図 1 7】

更新後のメソッド情報テーブルの例を示す図である。

【図 1 8】

更新後のメソッドパラメータテーブルの例を示す図である。

【図 1 9】

テストパターン自動引継処理の基本的なメインのフローチャートである。

【図 2 0】

不一致パラメータ引継処理を示すフローチャートである。

【図 2 1】

パラメータ名が一致しパラメータ型が不一致の場合の処理を示すフローチャートである。

【図 2 2】

判定処理とメソッドの属性との関係を示す図である。

【図 2 3】

メソッドの変更に応じたテストパターンの自動反映結果の一例を示す図である。

【図 2 4】

第 1 のメソッドの修正内容を示す図である。

【図 2 5】

第 1 のメソッドの属性毎の比較結果を示す図である。

【図 2 6】

第 2 のメソッドの修正内容を示す図である。

【図 2 7】

第 2 のメソッドの属性毎の比較結果を示す図である。

【図 2 8】

更新後の第 2 のメソッドのテストパターンの例を示す図である。

【図 2 9】

第 3 のメソッドの修正内容を示す図である。

【図 3 0】

第3のメソッドの属性毎の比較結果を示す図である。

【図31】

更新後の第3のメソッドのテストパターンの例を示す図である。

【図32】

第4のメソッドの修正内容を示す図である。

【図33】

第4のメソッドの属性毎の比較結果を示す図である。

【図34】

更新後の第4のメソッドのテストパターンの例を示す図である。

【図35】

第5のメソッドの修正内容を示す図である。

【図36】

第5のメソッドの属性毎の比較結果を示す図である。

【図37】

更新後の第5のメソッドのテストパターンの例を示す図である。

【図38】

引継対象メソッド指定画面の例を示す図である。

【図39】

テストパターン引継結果確認画面の例を示す図である。

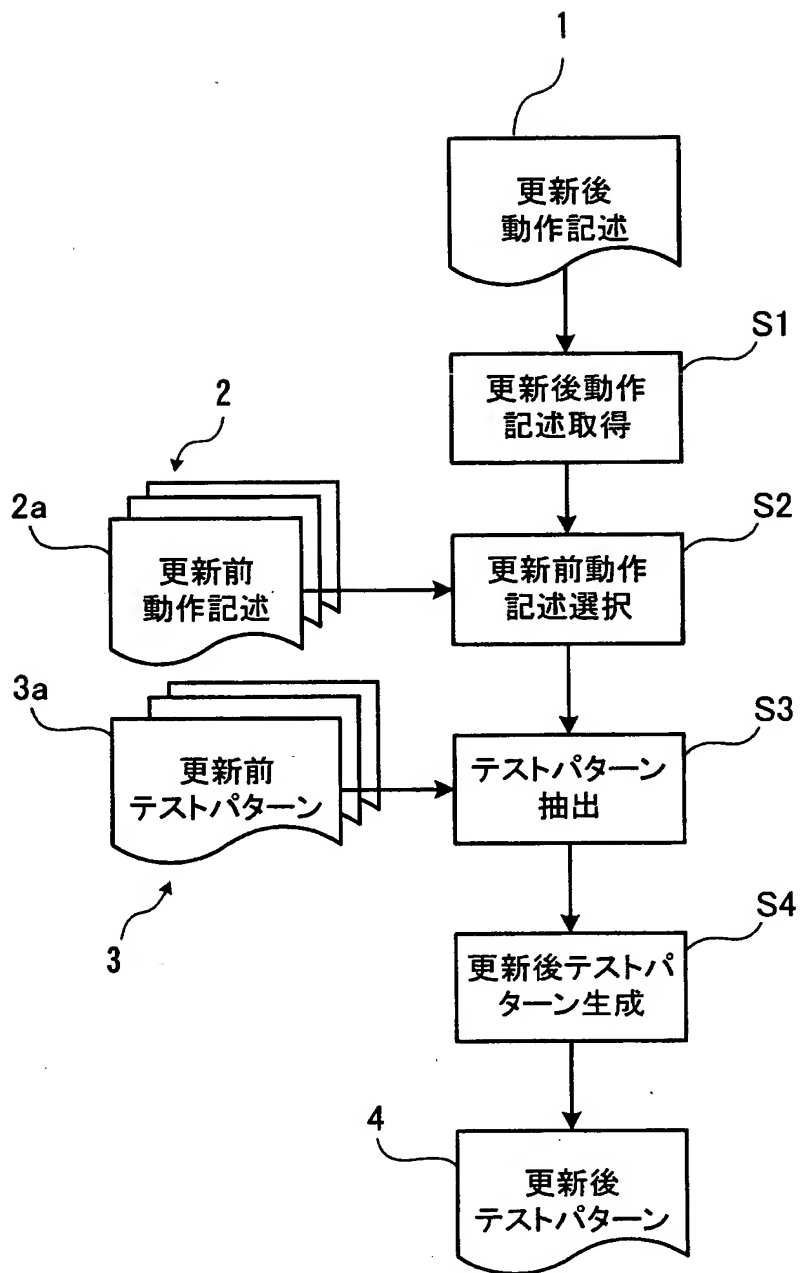
【符号の説明】

- 1 更新後動作記述
- 2 更新前動作記述
- 3 更新前テストパターン
- 4 更新後テストパターン
- 100 テスト支援装置
- 110 テストクラス記憶部
- 120 構文・意味解析部
- 130 第1の解析結果記憶部
- 140 テストパターン作成部

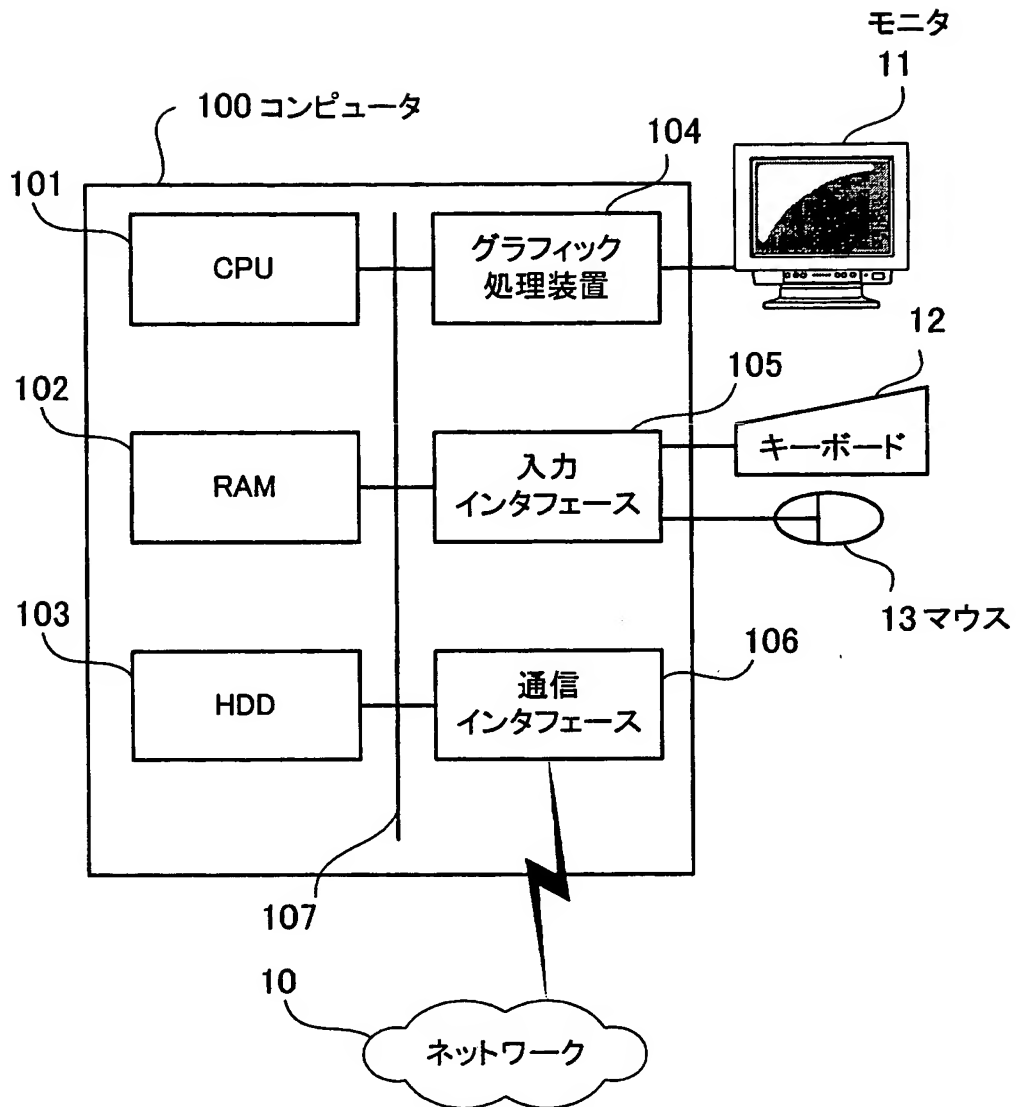
- 1 5 0 テストパターン記憶部
- 1 6 0 第 2 の解析結果記憶部
- 1 7 0 更新時自動引継ぎ部
- 1 8 0 テスト実施部
- 1 9 0 手動引継ぎ部

【書類名】 図面

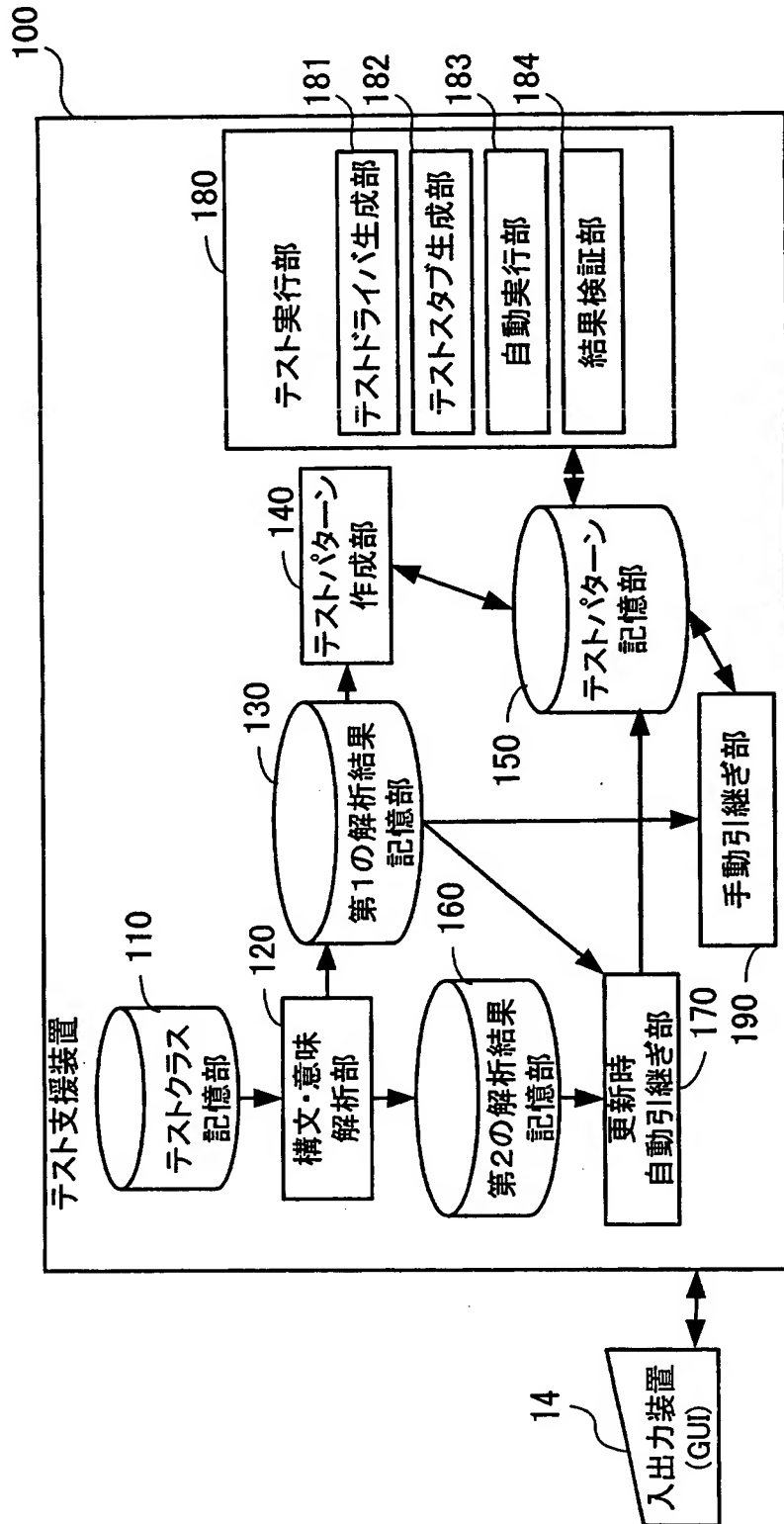
【図 1】



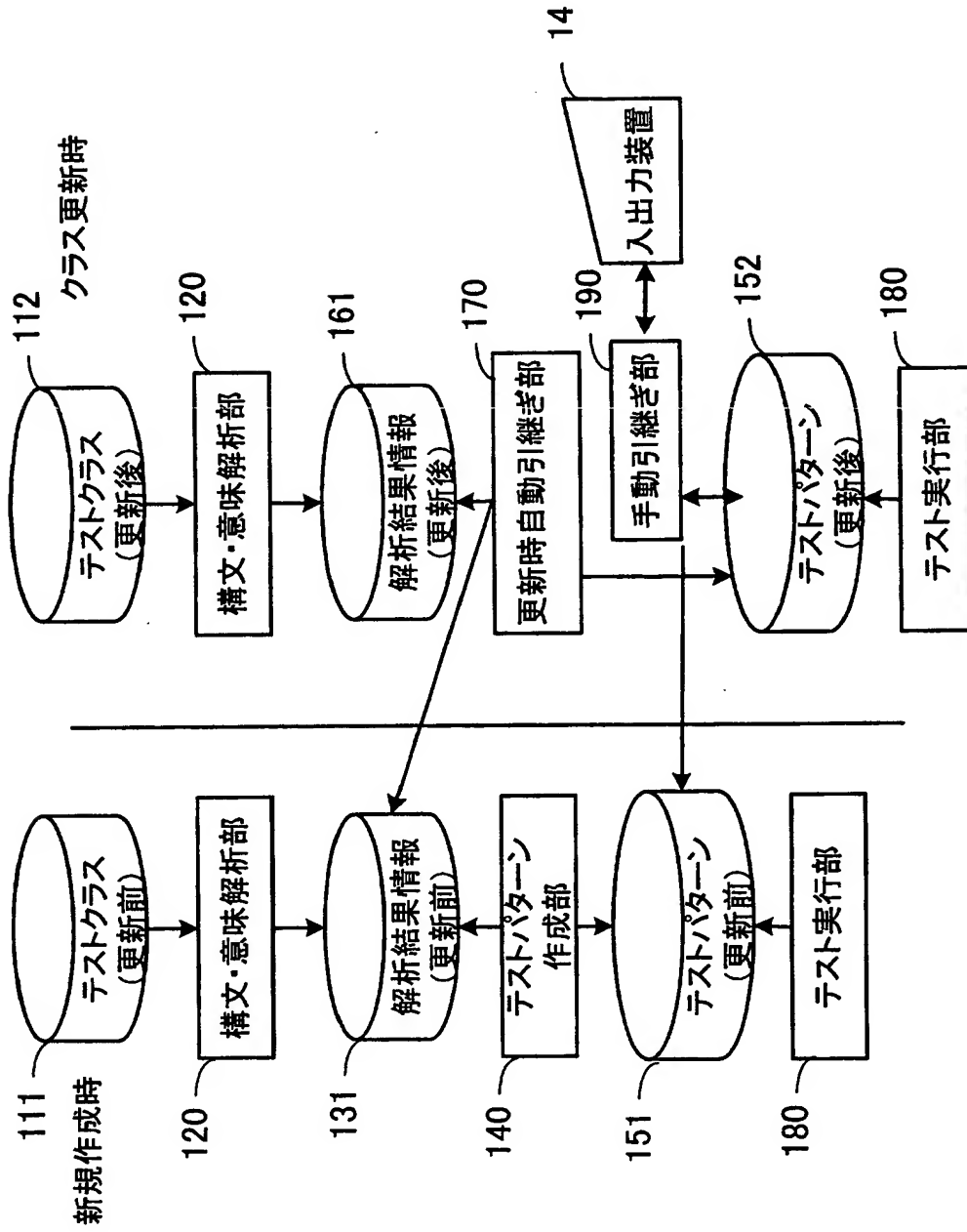
【図 2】



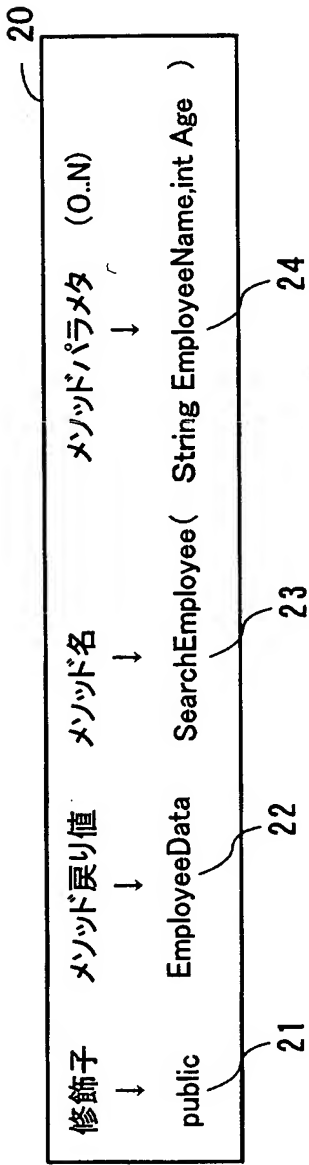
【図3】



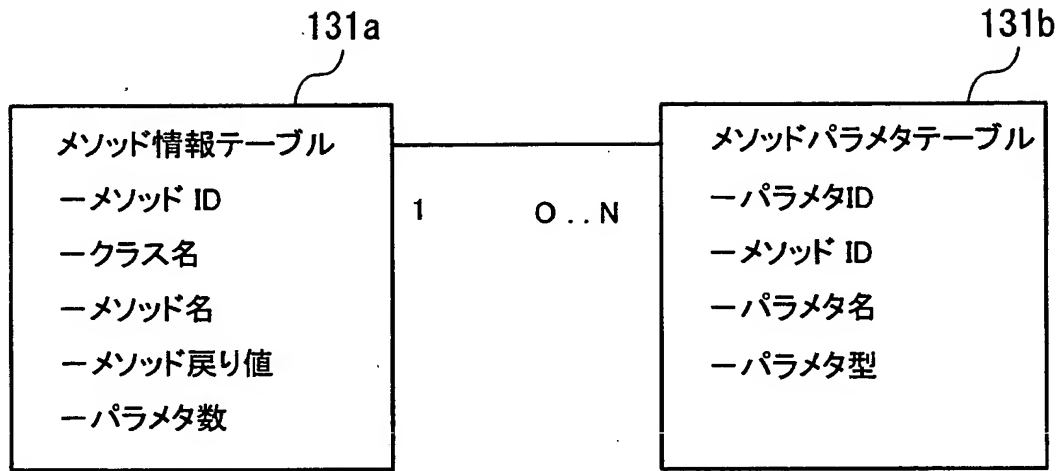
【図 4】



【図 5】



【図 6】



【図 7】

111

```
public class Employee extends Super TestClass{  
:  
// メソッド1 従業員名からの従業員検索機能。 111a  
public EmployeeData searchEmployee(String EmployeeName){  
:  
}  
// メソッド2 従業員氏名、年齢からの従業員検索機能。 111b  
public EmployeeData searchEmployee(String EmployeeName,int age,int code){  
:  
}  
// メソッド3 従業員氏名、年齢、部門コードからの従業員検索機能。 111c  
public EmployeeData searchEmployee(String EmployeeName,int Age,int code){  
:  
}  
// メソッド4 従業員氏名、部門コードからの従業員検索機能。 111d  
public EmployeeData searchEmployee(String EmployeeName,int code){  
:  
}  
// メソッド5 従業員コードからの従業員情報の更新機能。 111e  
public boolean updateEmployee(String EmployeeName,int age,int code){  
:  
}  
// メソッド6 従業員コードからの従業員情報の更新機能。  
public int deleteEmployee(int EmployeeNo){ 111f  
:  
}  
}
```

【図 8】

131a

メソッドID	クラス名	メソッド名	戻り値型	パラメタ数
1	Employee	searchEmployee	EmployeeData	1
2	Employee	searchEmployee	EmployeeData	2
3	Employee	searchEmployee	EmployeeData	3
4	Employee	searchEmployee	EmployeeData	2
5	Employee	updateEmployee	boolean	3
6	Employee	deleteEmployee	int	1

【図 9】

131b

パラメタ ID	メソッドID	パラメタ名	パラメタ名
1	1	EmployeeName	String
2	2	EmployeeName	String
3	2	age	int
4	3	EmployeeName	String
5	3	age	int
6	3	code	int
7	4	EmployeeName	String
8	4	code	int
9	5	EmployeeName	String
10	5	age	int
11	5	code	int
12	6	EmployeeNo	int

【図 1 0】

151a

No	メソッド名	パラメタ	正常 /異常	戻り値	テストの観点
M1	searchEmployee	EmployeeName		Employee Data	
M1-1	searchEmployee	“F通 太郎”	正常	検索結果	正常系のテスト
M1-2	searchEmployee	“”	異常	Null	氏名が空白の場合の異常系
M1-3	searchEmployee	Null	異常	Null	氏名がNullの場合の異常系

【図 1 1】

151b

No	メソッド 名	パラメタ		正常 /異常	戻り値	テストの観点
		EmployeeName	Age			
M2	searchEmployee	EmployeeName	Age		Employee Data	
M2-1	searchEmployee	“F通 太郎”	29	正常	検索結果	正常系のテスト
M2-2	searchEmployee	“”	29	異常	Null	氏名が空白の場合の異常系
M2-3	searchEmployee	Null	29	異常	Null	氏名がNullの場合の異常系
M2-4	searchEmployee	“F通 花子”	0	異常	Null	年齢が0の場合の異常系
M2-5	searchEmployee	“F通 花子”	10	異常	Null	年齢が従業員雇用範囲外の場合
M2-6	searchEmployee	“F通 花子”	65	異常	Null	年齢が従業員雇用範囲外の場合
M2-7	searchEmployee	“F通 花子”	Null	異常	Null	年齢がNullの場合

【図 1 2】

151c

No	メソッド名	パラメタ	正常 /異常			戻り値	テストの観点
M3	searchEmployee	EmployeeName	Age	code			
M3-1	searchEmployee	"F通 太郎"	29	12345	正常	検索結果	正常系のテスト
M3-2	searchEmployee	" "	29	12345	異常	Null	氏名が空白の場合の異常
M3-3	searchEmployee	Null	29	12345	異常	Null	氏名がNullの場合の異常
M3-4	searchEmployee	"F通 花子"	0	12345	異常	Null	年齢が0の場合の異常
M3-5	searchEmployee	"F通 花子"	10	12345	異常	Null	年齢が従業員雇用範囲外の場合
M3-6	searchEmployee	"F通 花子"	65	12345	異常	Null	年齢が従業員雇用範囲外の場合
M3-7	searchEmployee	"F通 花子"	Null	12345	異常	Null	年齢がNullの場合
M3-8	searchEmployee	"F通 花子"	29	0	異常	Null	従業員コードが0の場合
M3-9	searchEmployee	"F通 花子"	29	Null	異常	Null	従業員コードがNullの場合
M3-10	searchEmployee	Null	Null	Null	異常	Null	氏名、年齢がNullの場合

【図13】

151d

No	メソッド名	パラメタ	正常 /異常	戻り値	テストの観点
M4-1	searchEmployee	EmployeeName code			
M4-1	searchEmployee	"F通 太郎"	正常	検索結果	正常系のテスト
M4-2	searchEmployee	" "	異常	Null	氏名が空白の場合の異常系
M4-3	searchEmployee	Null	異常	Null	氏名がNullの場合の異常系
M4-4	searchEmployee	"F通 花子"	異常	Null	部コードが0の場合
M4-5	searchEmployee	"F通 花子"	異常	Null	部コードがNullの場合
M4-6	searchEmployee	Null	異常	Null	氏名、年齢がNullの場合

【図 14】

151e

No	メソッド名	パラメタ	正常 /異常	戻り値	テストの観点
M5	updateEmployee	EmployeeName			
		Age	code		
M5-1	updateEmployee	"F通 太郎"	29	12345	正常系のテスト
M5-2	updateEmployee	" "	29	12345	氏名が空白の場合の異常系
M5-3	updateEmployee	Null	29	12345	氏名がNullの場合の異常系
M5-4	updateEmployee	"F通 花子"	0	12345	年齢が0の場合の異常系
M5-5	updateEmployee	"F通 花子"	10	12345	年齢が従業員雇用範囲外の場合
M5-6	updateEmployee	"F通 花子"	65	12345	年齢が従業員雇用範囲外の場合
M5-7	updateEmployee	"F通 花子"	Null	12345	年齢がNullの場合
M5-8	updateEmployee	"F通 花子"	29	0	従業員コードが0の場合
M5-9	updateEmployee	"F通 花子"	29	Null	従業員コードがNullの場合
M5-10	updateEmployee	Null	Null	Null	従業員コード、氏名、年齢がNullの場合

【図 1 5】

151f

No	メソッド名	パラメタ	正常 /異常	戻り値	テストの観点
M6	deleteEmployee	code		Boolean	
M6-1	deleteEmployee	12345	正常	True	正常系のテスト
M6-2	deleteEmployee	0	異常	False	従業員コードが0の場合の 異常系
M6-3	deleteEmployee	Null	異常	False	従業員コードがNullの場合 の異常系
M6-4	deleteEmployee	1111	異常	False	従業員コードが該当なしの 場合の異常系

【図 16】

112

```
public class Employee extends SuperTestClass{  
    // メソッド1 従業員名からの従業員検索機能。 112a  
    private EmployeeData searchEmployee(String EmployeeName){  
        :  
    }  
    // メソッド2 従業員氏名、年齢からの従業員検索機能。 112b  
    public EmployeeData[] searchEmployee(String EmployeeName,int age,){  
        :  
    }  
    // メソッド3 従業員氏名、年齢、部門コードからの従業員検索機能。 112c  
    public EmployeeData searchEmployee(String EmployeeName,long Age,int  
    section,Boolean retire){  
        :  
    }  
    // メソッド4 従業員氏名、部門コードからの従業員検索機能。 112d  
    public EmployeeData[] searchEmployee(String EmployeeName,int section){  
        :  
    }  
    // メソッド5 従業員コードからの従業員情報の更新機能。 112e  
    public boolean updateEmployeebyNo(String EmployeeName,int age,int code){  
        :  
    }  
    // メソッド6 従業員コードからの従業員情報の更新機能。 112f  
    public int deleteEmployeebyNo(int EmployeeNo,StringEmployeeName){  
        :  
    }  
}
```

【図 1 7】

161a

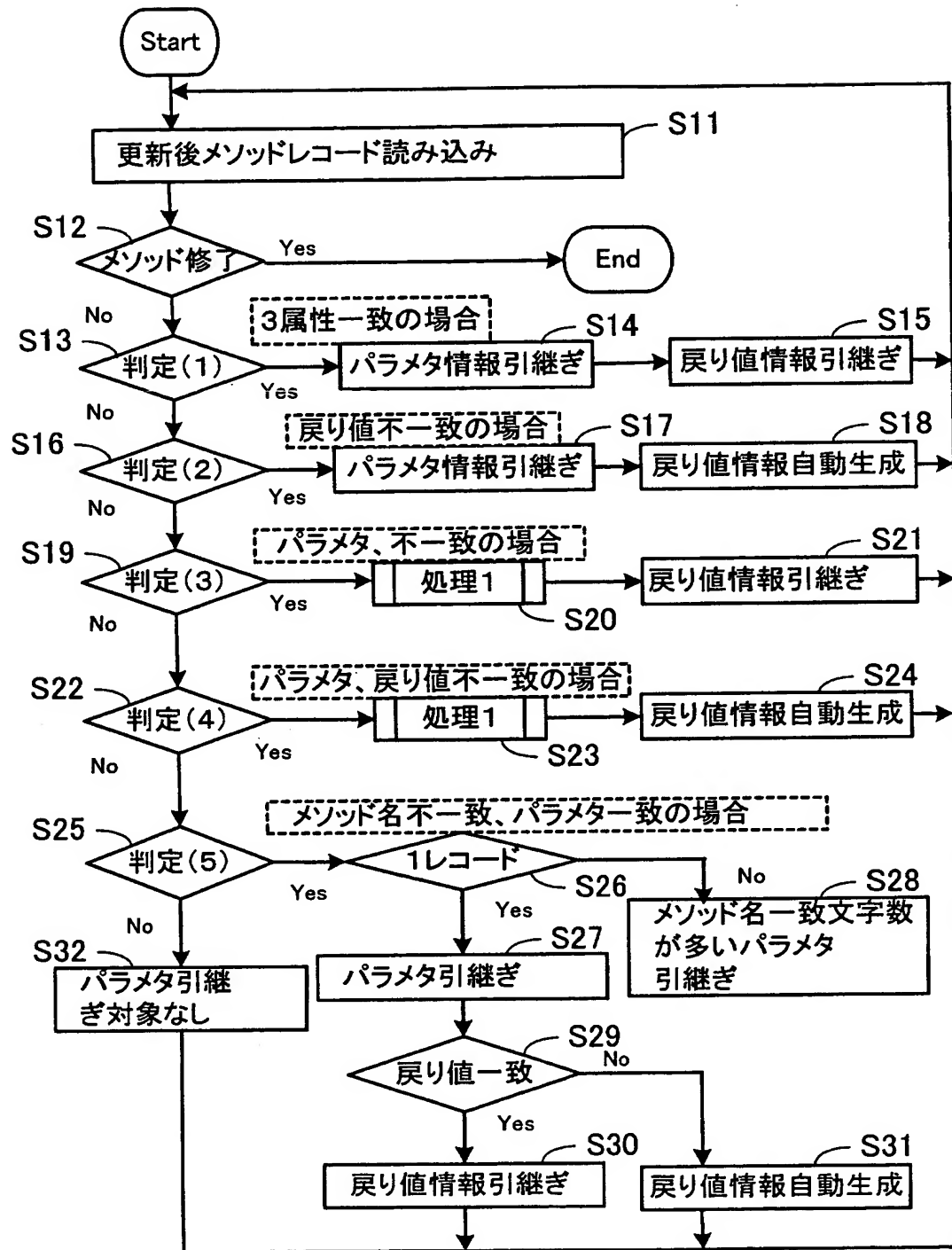
メソッド ID	クラス名	メソッド名	戻り値型	パラメタ 数
1	Employee	searchEmployee	EmployeeData	1
2	Employee	searchEmployee	EmployeeData[]	2
3	Employee	searchEmployee	EmployeeData	4
4	Employee	searchEmployee	EmployeeData[]	2
5	Employee	updateEmployeebyNo	Boolean	3
6	Employee	deleteEmployeebyNo	int	1

【図 1 8】

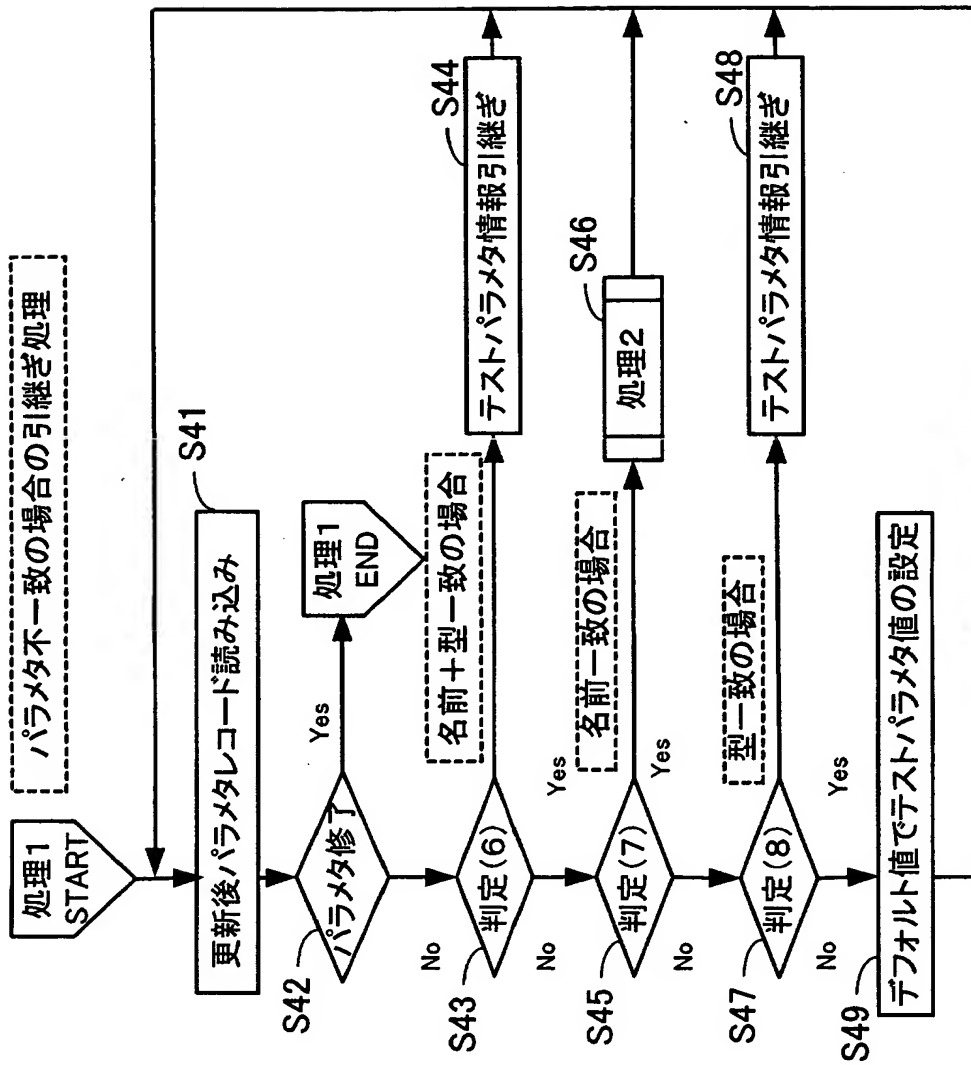
161b

パラメタ ID	メソッド ID	パラメタ名	パラメタ名
1	1	EmployeeName	String
2	2	EmployeeName	String
3	2	Age	int
4	3	EmployeeName	String
5	3	Age	Long
6	3	Section	int
7	3	Retire	Boolean
8	4	EmployeeName	String
9	4	Section	int
10	5	EmployeeName	String
11	5	Age	int
12	5	Code	int
13	6	EmployeeNo	int
14	6	EmployeeName	String

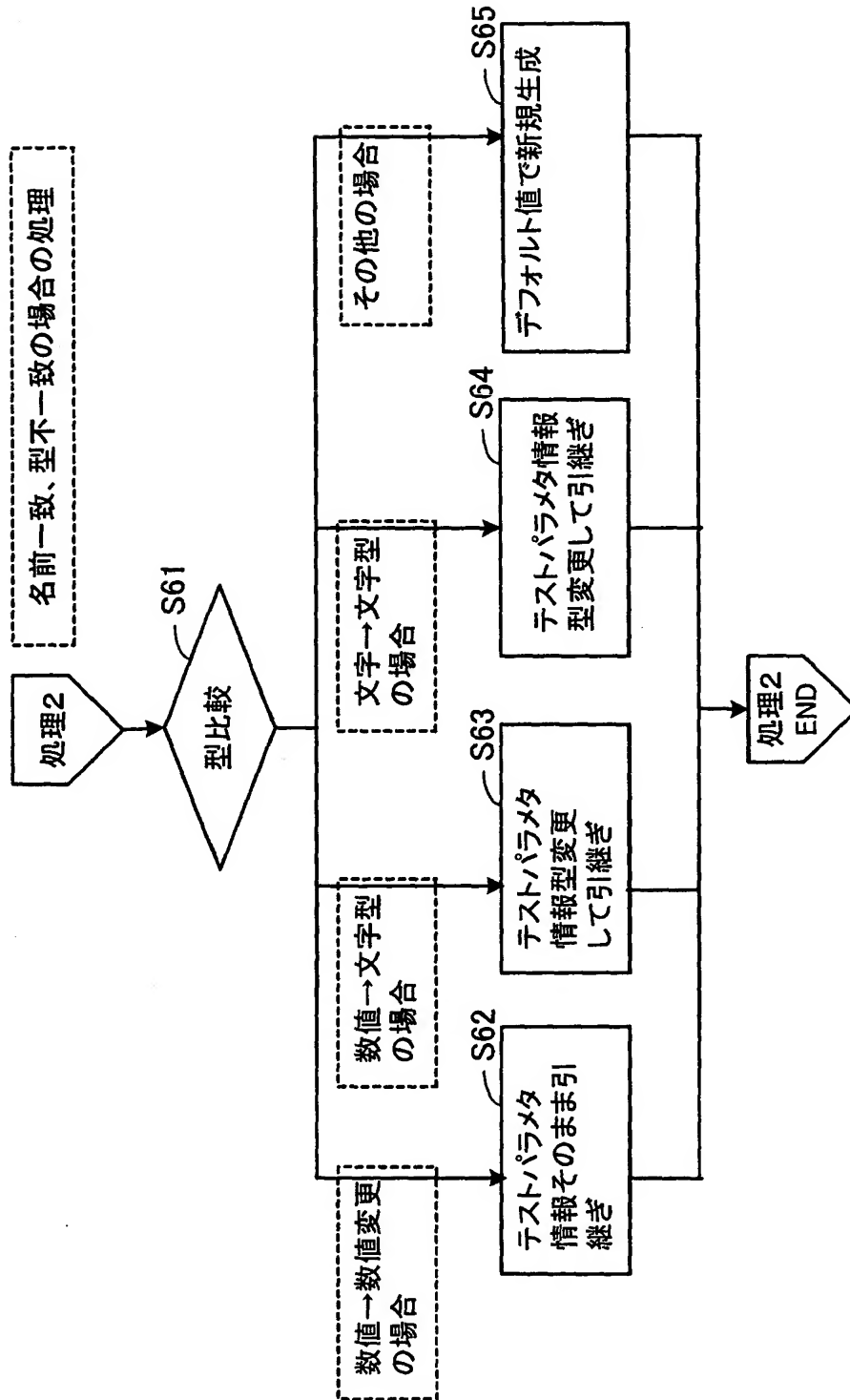
【図 19】



【図 2 0】



【図 21】



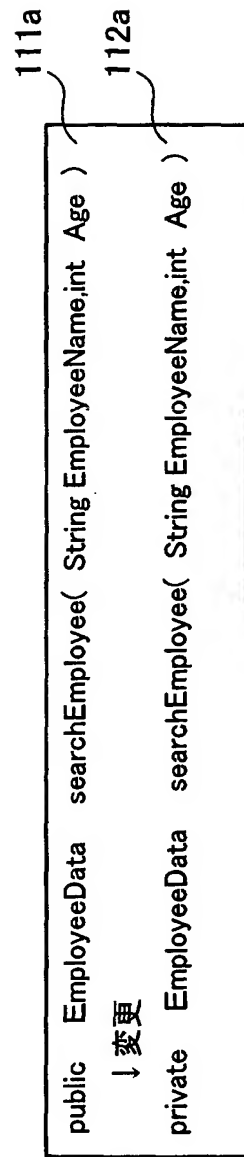
【図 2 2】

	メソッド名	パラメタ	戻り値
第1の判定	○	○	○
第2の判定	○	○	×
第3の判定	○	×	○
第4の判定	○	×	×
第5の判定	×	○	○
	×	○	×

【図23】

例名	判定	引き継がれたテストケース	引き継がれなかったテストケース
メソッド1	判定1:3属性完全一致	M1-1～M1-3	なし
メソッド2	判定2:戻値不一致	M2-1～M2-7	なし
メソッド3	判定3: パラメタ不一致	M3-1～M3-10	なし
メソッド4	判定4: パラメタ、戻り値 不一致	M4-1～M4-6	なし
メソッド5	判定5: パラメタ名 不一致	M5-1～M5-10	なし
メソッド6	判定で引き継ぎできず	なし	M6-1～M6-4

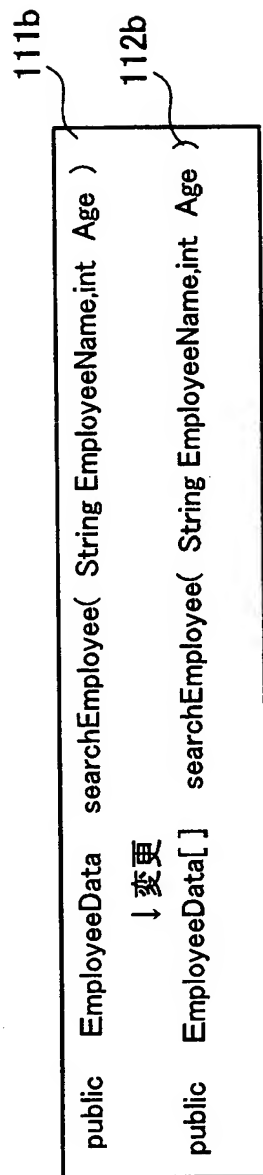
【図 2 4】



【図 25】

	メソッド名	パラメタ	戻り値	その他
変更前	searchEmployee	String EmployeeName, int Age	EmployeeData	Public
変更後	searchEmployee	String EmployeeName, int Age	EmployeeData	private

【図 2 6】



【図 2 7】

	メソッド名	パラメタ	戻り値	その他
変更前	searchEmployee	String EmployeeName,	EmployeeData	Public
変更後	searchEmployee	String EmployeeName,	EmployeeData[]	public

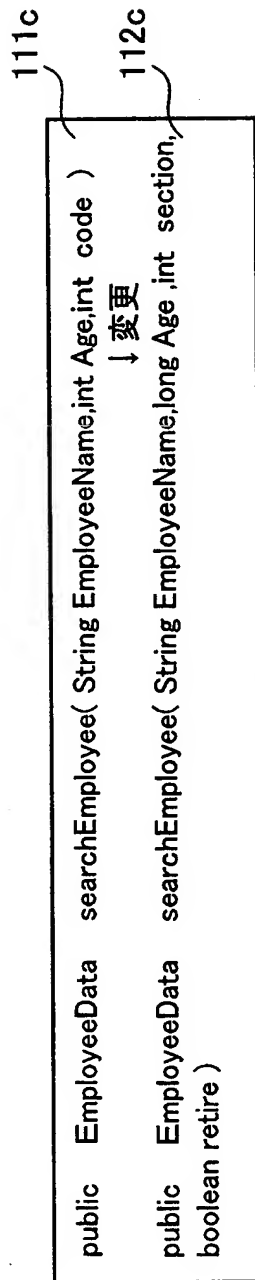
【図28】

152b

↓デフォルト値(Null)で設定

No	メソッド名	パラメタ		正常 /異常	戻り値	テストの観点
		EmployeeName	Age			
M2-1	searchEmployee	"F通 太郎"	29	正常	検索結果	正常系のテスト
M2-2	searchEmployee	" "	29	異常	Null	氏名が空白の場合の異常系
M2-3	searchEmployee	Null	29	異常	Null	氏名がnullの場合の異常系
M2-4	searchEmployee	"F通 花子"	0	異常	Null	年齢が0の場合の異常系
M2-5	searchEmployee	"F通 花子"	10	異常	Null	年齢が従業員雇用範囲外の場合
M2-6	searchEmployee	"F通 花子"	65	異常	Null	年齢が従業員雇用範囲外の場合
M2-7	searchEmployee	"F通 花子"	Null	異常	Null	年齢がnullの場合

【図 2 9】



【図 3 0】

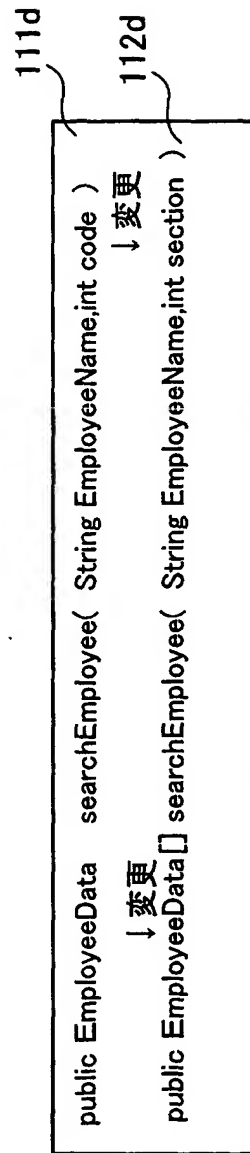
	メソッド名	パラメタ	戻り値	その他
変更前	searchEmployee	String EmployeeName, int Age, int code	EmployeeData	Public
変更後	searchEmployee	StringEmployeeName, long Age,int section, boolean retire	EmployeeData	Public

【図31】

152c

No.	メソッドパラメタ				正常/ 異常	戻り値	テスト観点
前	EmployeeName (String)	Age (int)	Code(int)	なし			
判定	①両方一致	②名前一致..	③型一致	④不一致(デフォルト生成)			
後	EmployeeName (String)	Age (long)	Section (int)	Retire (Boolean)			
M3-1	"F通 太郎"	29	12345	true	正常	検索結果	正常系のテスト
M3-2	"	29	12345		異常	Null	氏名が空白の場合の異常系
M3-3	Null	29	12345		異常	Null	氏名がnullの場合の異常系
M3-4	"F通 花子"	0	12345		異常	Null	年齢が0の場合の異常系
M3-5	"F通 花子"	10	12345		異常	Null	年齢が従業員雇用範囲外の場合
M3-6	"F通 花子"	65	12345		異常	Null	年齢が従業員雇用範囲外の場合
M3-7	"F通 花子"	Null	12345		異常	Null	年齢がnullの場合
M3-8	"F通 花子"	29	0		異常	Null	部コードが0の場合
M3-9	"F通 花子"	29	Null		異常	Null	部コードがnullの場合
M3-10	Null	Null/0			異常	Null	氏名、年齢がnullの場合

【図 32】



【図 3 3】

変更前	メソッド名	パラメタ	戻り値	その他
	searchEmployee	String EmployeeName, int code	EmployeeData	Public
変更後	searchEmployee	String EmployeeName, int section	EmployeeData□	Public

【図34】

—152d

No	メソッドパラメタ		正常 異常	戻り値	テスト観点
前	EmployeeName (String)			EmployeeData	
判定	①両方一致	③型一致			
後	EmployeeName (String)	Section (int)		EmployeeData []	
M4-1	"F通 太郎"	12345	正常	Null	正常系のテスト
M4-2	" "	12345	異常	Null	氏名が空白の場合の異常系
M4-3	Null	12345	異常	Null	氏名がNullの場合の異常系
M4-4	"F通 花子"	0	異常	Null	部コードが0の場合
M4-9	"F通 花子"	Null	異常	Null	部コードがNullの場合
M4-6	Null	Null	異常	Null	氏名、年齢がNullの場合

【図 3 5】

111e

112e

public EmployeeData updateEmployee(String EmployeeName,long Age,int code)

↓ 変更

public EmployeeData updateEmployeebyNo(String EmployeeName,long Age,int code)

【図 3 6】

	メソッド名	パラメタ	戻り値	その他
変更前	updateEmployee	String EmployeeName, long Age, int code	EmployeeData	Public
変更後	updateEmployeebyNo	String EmployeeName, long Age, int code	EmployeeData	Public

【図37】

—152e—

No	メソッド名	パラメタ	正常 /異常	戻り値	テストの観点
	updateEmployee ebyNo	EmployeeName Age code			
M5-1	updateEmployee ebyNo	"F通 太郎"	正常	検索結果	正常系のテスト
M5-2	updateEmployee ebyNo	" "	異常	Null	氏名が空白の場合の異常
M5-3	updateEmployee ebyNo	Null	異常	Null	氏名がNullの場合の異常系
M5-4	updateEmployee ebyNo	"F通 花子"	異常	Null	年齢が0の場合の異常系
M5-5	updateEmployee ebyNo	"F通 花子"	異常	Null	年齢が従業員雇用範囲外 の場合
M5-6	updateEmployee ebyNo	"F通 花子"	異常	Null	年齢が従業員雇用範囲外 の場合
M5-7	updateEmployee ebyNo	"F通 花子"	異常	Null	年齢がNullの場合
M5-8	updateEmployee ebyNo	"F通 花子"	異常	Null	従業員コードが0の場合
M5-9	updateEmployee ebyNo	"F通 花子"	異常	Null	従業員コードがNullの場合
M5-10	updateEmployee ebyNo	Null	異常	Null	従業員コード、氏名、年齢 がNullの場合

【図 3 8】

200
メソッド引継ぎ確認
TestSrc_01.java
が更新されています。メソッドを対応付け、引継ぐメソッドを確定してください。
更新後のメソッド情報(I): 210 更新前のメソッド情報(E): 220

クラス名(新)	メソッド名(新)	▲クラス名(旧)	メソッド名(新旧)	ファイル名(旧)
AAA	MethodA10	AAA	MethodA10	TestSrc_03.java
AAA	MethodA20	AAA	MethodA2(int A)	TestSrc_03.java
AAA	MethodA30	AAA	MethodA3(int B)	TestSrc_02.java
AAA	MethodA40			
AAA	MethodA50			
AAA	MethodA60			
AAA	MethodA70			
AAA	MethodA80			
AAA	MethodA90			
AAA	MethodAA0			

201 上へ(U)
202 下へ(L)
203 削除(D)

204 追加(A)

230 引継ぎ候補のないメソッド一覧(M):

クラス名	メソッド名	ファイル名
AAA	MethodB10	TestSrc_1.java

205 OK 206 キャンセル 207 ヘルプ

右の『引継ぎ候補のないメソッド一覧』
リストより引継ぎ対象となるメソッドを
選択し、右上の『変更前のメソッド情報』
リストに登録してください。

【图 39】

300

テストケース編集-Class1::Method1(PureJava アプリケーション):1

☐ ☐ ☐ ☒

310

テストケース

テストケース番号	テスト内容	正常/異常	実施方...	注目因子タイプ	メソッドパラメタ名	変数名
<input checked="" type="checkbox"/> Method1			.			
<input checked="" type="checkbox"/> Method1_001	テスト内容	正常	▼ .テストの...	param1	▼	
<input checked="" type="checkbox"/> Method1_002	テスト内容	正常	▼ .	param3	▼	
<input checked="" type="checkbox"/> Method1_003	テスト内容	正常	▼ .	param2	▼	
			.			
			.			
			.			
			.			
			.			
<◀			.			▶

320

パラメタデータ 変数データ

パラメタ	属性	値
<input checked="" type="checkbox"/> param1	int	10
<input checked="" type="checkbox"/> param2	MyClass	
<input checked="" type="checkbox"/> field1	int[]	-
<input checked="" type="checkbox"/> bbb	ArrayList	-
<input checked="" type="checkbox"/> tblField	hashmap	-
<input checked="" type="checkbox"/> param3	boolean	false

【書類名】 要約書

【要約】

【課題】 テスト対象のプログラムの内容が更新された場合に、更新後のプログラム用のテストパターンを効率的に作成できるようにする。

【解決手段】 開発対象プログラムの内容が更新されたとき、更新後の開発対象プログラムに含まれる機能の動作内容を定義した更新後動作記述 1 を取得する（ステップ S 1）。次に、更新前の開発対象プログラムに含まれる機能の動作内容を定義した複数の更新前動作記述 2 から、更新後動作記述と共通性の高い更新前動作記述 2 a を選択し（ステップ S 2）、更新前の開発対象プログラムの動作テストのために作成された複数の更新前テストパターン 3 から、選択された更新前動作記述の動作テストのための更新前テストパターン 3 a を抽出し（ステップ S 3）、抽出した更新前テストパターン 3 a の少なくとも一部を引き継いで、更新後動作記述 1 の動作テストのための更新後テストパターン 4 を生成する（ステップ S 4）。

【選択図】 図 1

出 願 人 履 歴 情 報

識別番号 [000005223]

1. 変更年月日 1996年 3月26日

[変更理由] 住所変更

住 所 神奈川県川崎市中原区上小田中4丁目1番1号
氏 名 富士通株式会社